CAMPUS
DE EXCELENCIA
INTERNACIONAL

Graduado en Ingeniería Informática

Universidad Politécnica de Madrid

Facultad de Informática

TRABAJO FIN DE GRADO

Sistema de recomendación semántico para la movilidad en el entorno AFRICA BUILD Portal

Autor: Carlos Muiño Migallón

Director: Víctor Maojo García

Cotutor: Máximo Ramírez Robles

MADRID, JUNIO DE 2013

A mi familia y amigos.

RESUMEN

La Gestión de Recursos Humanos a través de Internet es un problema latente y presente actualmente en cualquier sitio web dedicado a la búsqueda de empleo. Este problema también está presente en AFRICA BUILD Portal. AFRICA BUILD Portal es una emergente red socio-profesional nacida con el ánimo de crear comunidades virtuales que fomenten la educación e investigación en el área de la salud en países africanos. Uno de los métodos para fomentar la educación e investigación es mediante la movilidad de estudiantes e investigadores entre instituciones, apareciendo así, el citado problema de la gestión de recursos humanos. Por tanto, este trabajo se centra en solventar el problema de la gestión de recursos humanos en el entorno específico de AFRICA BUILD Portal.

Para solventar este problema, el objetivo es desarrollar un sistema de recomendación que ayude en la gestión de recursos humanos en lo que concierne a la selección de las mejores ofertas y demandas de movilidad. Caracterizando al sistema de recomendación como un sistema semántico el cual ofrecerá las recomendaciones basándose en las reglas y restricciones impuestas por el dominio.

La aproximación propuesta se basa en seguir el enfoque de los sistemas de Matchmaking semánticos. Siguiendo este enfoque, por un lado, se ha empleado un razonador de lógica descriptiva que ofrece inferencias útiles en el cálculo de las recomendaciones y por otro lado, herramientas de procesamiento de lenguaje natural para dar soporte al proceso de recomendación. Finalmente para la integración del sistema de recomendación con AFRICA BUILD Portal se han empleado diversas tecnologías web.

Los resultados del sistema basados en la comparación de recomendaciones creadas por el sistema y por usuarios reales han mostrado un funcionamiento y rendimiento aceptable. Empleando medidas de evaluación de sistemas de recuperación de información se ha obtenido una precisión media del sistema de un 52%, cifra satisfactoria tratándose de un sistema semántico.

Pudiendo concluir que con la solución implementada se ha construido un sistema estable y modular posibilitando: por un lado, una fácil evolución que debería ir encaminada a lograr un rendimiento mayor, incrementando su precisión y por otro lado, dejando abiertas nuevas vías de crecimiento orientadas a la explotación del potencial de AFRICA BUILD Portal mediante la Web 3.0.

ABSTRACT

The Human Resource Management through Internet is currently a latent problem shown in any employment website. This problem has also appeared in AFRICA BUILD Portal. AFRICA BUILD Portal is an emerging socio-professional network with the objective of creating virtual communities to foster the capacity for health research and education in African countries. One way to foster this capacity of research and education is through the mobility of students and researches between institutions, thus appearing the Human Resource Management problem. Therefore, this dissertation focuses on solving the Human Resource Management problem in the specific environment of AFRICA BUILD Portal.

To solve this problem, the objective is to develop a recommender system which assists the management of Human Resources with respect to the selection of the best mobility supplies and demands. The recommender system is a semantic system which will provide the recommendations according to the domain rules and restrictions.

The proposed approach is based on semantic matchmaking solutions. So, this approach on the one hand uses a Description Logics reasoning engine which provides useful inferences to the recommendation process and on the other hand uses Natural Language Processing techniques to support the recommendation process. Finally, Web technologies are used in order to integrate the recommendation system into AFRICA BUILD Portal.

The results of evaluating the system are based on the comparison between recommendations created by the system and by real users. These results have shown an acceptable behavior and performance. The average precision of the system has been obtained by evaluation measures for information retrieval systems, so the average precision of the system is at 52% which may be considered as a satisfactory result taking into account that the system is a semantic system.

To conclude, it could be stated that the implemented system is stable and modular. This fact on the one hand allows an easy evolution that should aim to achieve a higher performance by increasing its average precision and on the other hand keeps open new ways to increase the functionality of the system oriented to exploit the potential of AFRICA BUILD Portal through Web 3.0.

AGRADECIMIENTOS

Me gustaría comenzar dando las gracias a personas que de una u otra manera me han ayudado a realizar este proyecto y también durante toda la carrera.

En primer lugar, a mi tutor en este proyecto, Víctor Maojo García, por brindarme la oportunidad de trabajar en el Grupo de Informática Biomédica y por orientarme en la realización de este Trabajo.

A mis compañeros del proyecto AFRICA BUILD Ana, Maxi, Miguel e Inma. Destacando especialmente a Maxi quien ha tenido un papel importante en este Trabajo, siendo el cotutor, ofreciéndome su ayuda y consejos.

A algunos profesores que he tenido a lo largo de la carrera, destacando entre ellos a Miguel Ángel Pascual quien ha sido mi tutor curricular a lo largo de la carrera y a Xavier Ferré por haber confiado en mí para realizar varias becas.

A mis padres, hermanas y cuñados por ayudarme en todo lo posible durante todos estos años de formación y por estar siempre ahí.

A mis amigos de la universidad por las experiencias vividas juntos en estos años entre los que destaco a: Pablo, Lorenzo, Juli, Redondo, Garri, Miguel y Alberto.

A mis amigos de toda la vida, Miguel, Macias, Adry, Rafa, Samu, Rayo y Pablo por todos los momentos pasados juntos desde hace muchos años.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1

INTRODUCCIÓN

1.1 Planteamiento del problema

AFRICA BUILD es una acción coordinada entre distintas instituciones con el ánimo de apoyar y desarrollar avanzados centros de excelencia en la atención a la salud, educación e investigación en países Africanos, a través de las Tecnologías de la Información.

Dos importantes objetivos de AFRICA BUILD son, por un lado, el desarrollo de una infraestructura tecnológica colaborativa para ayudar a crear comunidades virtuales que fomenten la educación y la investigación en el área de la salud, derivando en la creación de la red socio-profesional AFRICA BUILD Portal; y por otro lado, otro de los objetivos es el de fomentar la movilidad de estudiantes y personal investigador entre instituciones.

Así, como intersección de estos dos objetivos, aparece la necesidad de desarrollar una herramienta integrada en AFRICA BUILD Portal, que respalde y ayude en la tarea de la gestión de la movilidad, encontrándonos con el problema de gestión de recursos humanos a través de Internet [1] [2].

A día de hoy, es un hecho que Internet y la Web experimentan un crecimiento vertiginoso a diario, en cuanto a la cantidad de recursos que ofrecen, provocando la necesidad de contar con herramientas que ayuden a filtrar la información de acuerdo a los requisitos y/o preferencias expresadas por los usuarios. Esta necesidad se acentúa más si se enmarca en el contexto de la gestión de recursos humanos donde nos encontramos ante una incesante cantidad de información en forma de ofertas y demandas de empleo [3].

Este contexto ha experimentado una evolución en cuanto al medio de comunicación y de hecho se encuentra todavía en ella, pasando de los tradicionales anuncios de empleo publicados en prensa, a la publicación en portales web de empleo y más recientemente en redes sociales. Particularmente este último medio es destacable, dado el auge de las redes sociales, estas se están convirtiendo en un complemento perfecto en los procesos

de selección, apareciendo así el término de "La Selección 2.0" que hace referencia a una nueva forma de llevar a cabo los procesos de selección. Mediante esta nueva selección, las empresas seleccionan sus empleados mediante la búsqueda de ellos en redes sociales, ya que estas ofrecen información de gran valor generada por el uso de la red, obteniendo así no solo las características específicas relacionadas a una demanda de trabajo, si no también poder conocer como es realmente el candidato en su día a día [4].

Pese a contar con estos nuevos paradigmas en la gestión de recursos humanos ya sea mediante portales web de empleo o búsqueda en redes sociales, el problema de filtrar información de cara a lo realmente buscado permanece latente.

1.2 Objetivos

Dado el problema presentado anteriormente y las soluciones actuales, encontramos como principal objetivo el desarrollo de un sistema de recomendación que ayude en la gestión de recursos humanos en lo que concierne a la selección de demandas con respecto a ofertas y viceversa, es decir, el sistema facilitará la búsqueda de información ofreciendo estas recomendaciones basadas en lo que el usuario está buscando y a partir de estas recomendaciones será el usuario quien tome cualquier decisión. Para lograr este principal objetivo el sistema debe permitir la introducción de ofertas y demandas de movilidad, que posteriormente serán cruzadas automáticamente.

Para conformar el sistema de recomendación se han marcado unos objetivos que debe cumplir, siendo los siguientes:

- Sistema de recomendación semántico.
  Las recomendaciones serán calculadas en función a una base de conocimiento que modelará el dominio de la movilidad en el área específica en el que se encuentra la herramienta, es decir, la salud y África.

- Clasificación de recomendaciones.
  Cada recomendación expresará el grado de proximidad entre la oferta y demanda de una manera cuantitativa de modo que todas las recomendaciones pertenecientes a una misma oferta/demanda puedan ser comparadas.

- Razonamiento a la recomendación.
  Además del grado de proximidad entre oferta y demanda cada recomendación deberá ofrecer una explicación en lenguaje natural a tal grado de proximidad.

- Recomendación dual.
  Las recomendaciones deberán ser calculadas desde los dos posibles puntos de vista, es decir, recomendaciones desde el punto de vista de la oferta y desde el de la demanda, ya que las recomendaciones no tienen por qué ser simétricas.

- Extracción de información a partir de lenguaje natural.
  Alguna información que será empleada para realizar la recomendación deberá ser extraída de partes, expresadas en lenguaje natural, de las ofertas y demandas.

- Integración en AFRICA BUILD Portal.
  El sistema de recomendación deberá estar integrado totalmente en la plataforma AFRICA BUILD Portal, entendiendo por esta integración total que el sistema sea una parte más de la plataforma; presentando una arquitectura y estilo, en lo que se refiere a su interfaz, acorde con la plataforma.

- Optimización.
  Dado que la plataforma AFRICA BUILD Portal será accedida por lo general desde dispositivos con prestaciones bastante reducidas, tanto en lo que respecta a capacidad de computo como de conectividad a Internet, el sistema deberá presentar características que permitan un funcionamiento lo más eficiente como sea posible.

1.3 Solución propuesta

Ante el problema y objetivos presentados anteriormente, nace el sistema denominado "AFRICA BUILD Portal - Mobility Brokerage Service" a partir de la adaptación, mejora y re-implementación del Mobility ideado en el proyecto INFOBIOMED [5]. A continuación en la figura 1 se muestra un esquema del funcionamiento general del sistema:

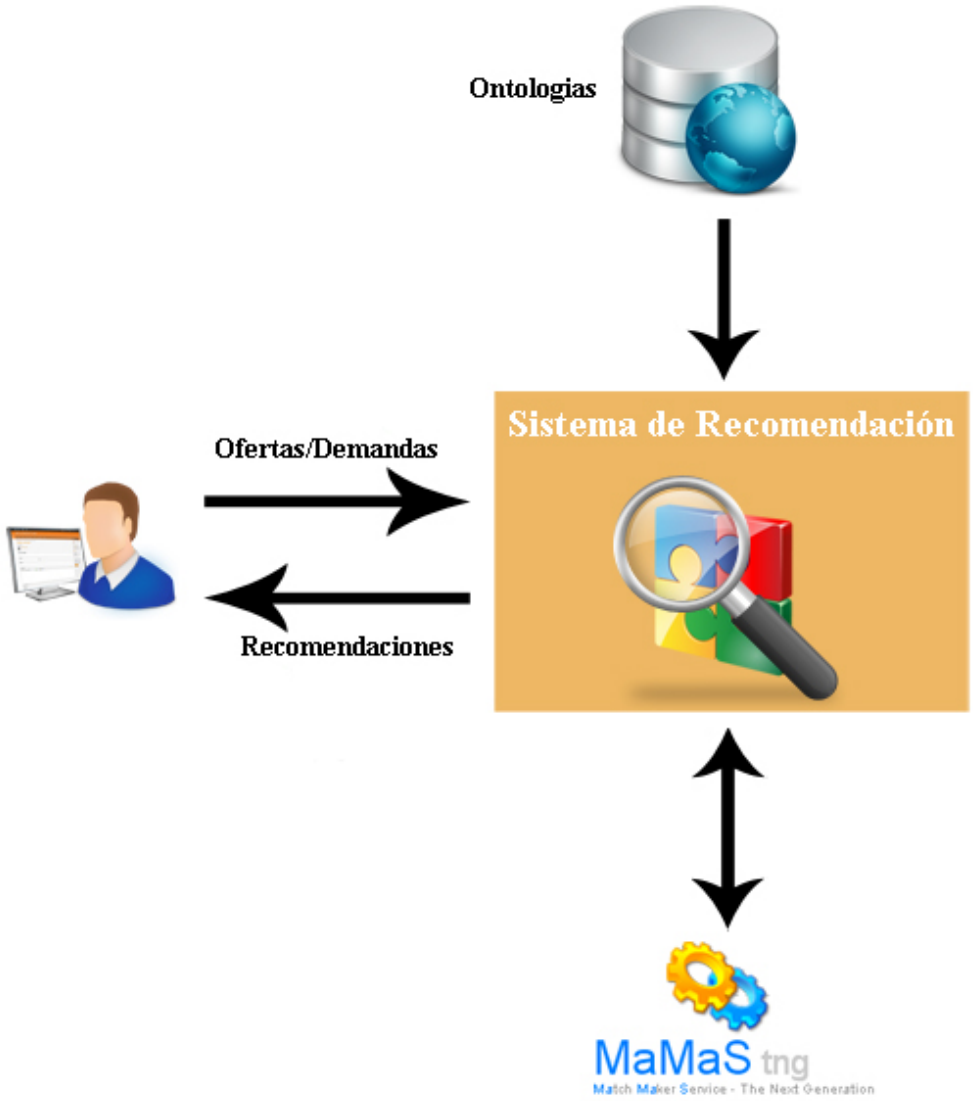


Figura 1-1: Visión general del sistema de recomendación

Este sistema se puede dividir en cuatro grandes componentes o bloques:

- Un plugin para AFRICA BUILD Portal que actuará de Interfaz gráfica y de sistema integrador con los otros módulos del sistema de recomendación. En AFRICA BUILD Portal todos los componentes son plugins, que no son más que módulos instalables y configurables individualmente. De esta manera se logra la integración del sistema de recomendación con AFRICA BUILD Portal, siendo un plugin más con la misma arquitectura y diseño que cualquier otro plugin ofreciendo así una interfaz gráfica para el sistema de recomendación, accedida desde cualquier navegador web. Siendo, de tal manera, el punto desde donde poder crear, editar o eliminar las ofertas y demandas y donde poder consultar las recomendaciones sobre estas ofertas y demandas.

- Un analizador sintáctico que formalizará las ofertas y demandas, insertadas a través del plugin para AFRICA BUILD Portal, escritas en Lenguaje Natural a un lenguaje lógico bajo el que operará el sistema de recomendación. Para ello realizará primeramente tareas de procesamiento de lenguaje natural como identificación sintáctica y semántica de elementos, corrección ortográfica o sinonimia y posteriormente una traducción a descripción lógica.

- Una serie de ontologías que actuarán como un vocabulario formal para modelar y describir el dominio de las ofertas y demandas y a partir de las cuales el sistema de recomendación calculará las recomendaciones. Para lograr una recomendación dual, es decir, desde los dos puntos de vista posibles, se tendrán duplicadas las ontologías presentando distintas definiciones de los mismos conceptos.

- El algoritmo de recomendación que cruzará, periódica y automáticamente, las ofertas y demandas existentes en el sistema y que previamente habrán sido formalizadas, las cuales posteriormente serán visibles desde AFRICA BUILD Portal. Las recomendaciones serán calculadas mediante inferencias lógicas provistas por un motor de razonamiento para lógica descriptiva denominado "MaMaS-tng".

Capítulo 2

ESTADO DE LA CUESTIÓN

2.1 Matchmaking

2.1.1 Introducción

El Matchmaking o los sistemas de Matchmaking son un tipo de agente software o sistema agente. Un agente software es un programa que actúa en nombre de un usuario, servicio u otro programa para realizar algún tipo de tarea u operación que el usuario, servicio o programa efectuarían. Las principales características de los agentes software son que se ejecutan automáticamente sin tener que ser invocados por un tercero, toman decisiones por ellos mismos sin la intervención humana y pueden actuar con otros agentes de manera coordinada. Los agentes software se clasifican en diferentes clases de agentes dependiendo de la funcionalidad y objetivo que tengan. Los agentes software se encuentran hoy en día en distintos entornos como tiendas o mercados electrónicos, sistemas de monitorización o en sistemas de conexión de redes y comunicaciones [6].

Volviendo al Matchmaking, el termino Matchmaking como tal proviene de culturas antiguas donde existía el personaje del Matchmaker, que era una persona que se encargaba de unir personas con el objetivo del matrimonio [7], de hecho, la traducción al español del término Matchmaking es "casamentero" y "actividades de casamiento" [8]. Sin embargo, actualmente el termino ha sido adoptado para dar nombre a este tipo de sistemas agente y no solo como se podría pensar para sistemas de citas online y similares.

Una primera definición algo general del término Matchmaking en este entorno es el de un sistema que se encarga de recuperar y clasificar descripciones de cualquier tipo de servicio o entidad que mejor coincidan con una petición dada [9]. Una mejor aproximación sería la de un sistema que actúa como facilitador inteligente de información para emparejar de manera colaborativa descripciones de productos o servicios, para una tarea, negociación o transacción de mercado, suministradas por proveedores y consumidores [10].

Para aproximar algo más la definición de Matchmaking hay que situar el ámbito de operación de estos sistemas. El Matchmaking normalmente se encuentra en entornos donde por un lado se ofrecen servicios o entidades y por otro lado se demandan esos mismos o similares servicios o entidades, como por ejemplo ocurre en el caso de los mercados electrónicos, donde por un lado se ofrecen bienes o servicios y por otro lado estos bienes y servicios son adquiridos, en los cuales estos sistemas juegan un rol importante [10] [11]; o en el caso de la búsqueda de empleo donde por un lado se tienen empresas o instituciones que ofrecen puestos de empleo y por otro lado personas que buscan o demandan un empleo. Además de en estos dos casos de ejemplo, el Matchmaking es extrapolable a cualquier caso en el que se tengan contrapartes teniendo por un lado demandas y por otro ofertas [11].Así, de manera genérica se puede concluir que un sistema de Matchmaking es un sistema que realiza una tarea de Recuperación de Información interpretada como una función que acepta como entrada un conjunto de consultas(demandas) y de recursos(ofertas) y devuelve como salida una lista clasificada con las k ofertas más apropiadas con respecto a la consulta o demanda, obtenidas mediante la similitud entre ellas [12].

2.1.2 Evolución de los sistemas de Matchmaking

El Matchmaking ha llegado a ser un área de extensa investigación con la eclosión de la Web Semántica, sin embargo los primeros sistemas denominados de Matchmaking datan de mucho más tiempo atrás. Estos primeros sistemas se basaban en dar respuesta a simples consultas realizadas sobre bases de datos lo que les difiere considerablemente de los sistemas de Matchmaking actuales [13]. A continuación se presenta la evolución de los sistemas de Matchmaking, diferenciándolos en tres tipos de sistemas:

- Sistemas de Matchmaking basados en bases de datos.

Los primeros sistemas de Matchmaking se desarrollan sobre bases de datos de conocimiento. Estos sistemas de Matchmaking trataron de superar las limitaciones que presentaban las bases de datos relacionales usando pesos atribuidos a variables de búsqueda. Continuando ese enfoque se realizaron distintas aproximaciones para tratar de extender el lenguaje SQL con cláusulas de "preferencia" con el fin de emparejar de manera débil la información de bases de datos estructuradas. Estos intentos desembocan en la invención de KQML que es un lenguaje y protocolo de manipulación y consulta de conocimiento que permite la comunicación entre agentes software y sistemas de conocimiento. Al mismo tiempo aparece KIF que es un lenguaje para el intercambio de conocimiento entre diferentes componentes software. Estas dos tecnologías marcaran las bases para los siguientes sistemas de Matchmaking [13].

- Sistemas de Matchmaking sintáctico.

Los siguientes sistemas de Matchmaking que rompen con las primeras aproximaciones basadas en bases de datos se denominan sistemas de Matchmaking sintáctico. Esta denominación es dada debido a que se basan en comparaciones sintácticas entre los elementos a emparejar [13].

La primera aproximación de este tipo es el sistema SHADE que emplea KQML Y KIF. El sistema operaba sobre un lenguaje de texto estructurado y reglas lógicas, aunque no había base de conocimiento ni inferencias. El emparejamiento se basaba en hacer coincidir la demanda y oferta mediante un algoritmo de unificación, similar al de Prolog, el cual buscaba cadenas lógicas formadas y mediante un patrón de expresiones regulares unificaba los términos para compararlos [9] [14].

Las siguientes aproximaciones se centraron en la computación de distancias a partir de vectores que representaban las descripciones. Esta computación de distancias es una tarea básica en Recuperación de Información que se basa en aplicar una función para obtener la similitud entre vectores ponderados. Estos vectores describen la oferta y demanda a partir de sus términos base que son atributos con valores atómicos [10] [13].

Siguiendo esta aproximación aparece el sistema COINS que opera sobre la información presente en texto libre. COINS convierte cualquier entrada de texto en su correspondiente vector que posteriormente es cortado mediante un análisis de frecuencias de sus conceptos. Conceptos que son buscados en un corpus, que es una base de datos que recopila conceptos frecuentes que aparecieron en anteriores consultas. Finalmente el emparejamiento es realizado comparando los vectores de las descripciones [9].

También siguiendo esta aproximación se encuentra el sistema GRAPPA, que emplea técnicas similares al anterior pero es mucho más potente. GRAPPA permite la generación de estructuras de ofertas y demandas correspondientes a cualquier dominio y permite definir sus propias métricas y funciones de distancia [9].

Además de estos sistemas presentados han sido desarrollados algunos otros menos relevantes como por ejemplo SIMS o InfoSleuth, sin embargo todos ellos presentan limitaciones en ciertos dominios y sus resultados se basan en simples comparaciones de texto o similares que resultan en un todo o nada, por ello los siguientes sistemas de Matchmaking toman un enfoque semántico [13].

- Sistemas de Matchmaking semántico.

Los siguientes sistemas de Matchmaking son los denominados sistemas semánticos. Estos sistemas buscan dar nuevas soluciones que no estén basadas en simples comparaciones que resultan en algo probabilístico [15]. Su visión es ofrecer la posibilidad de clasificar y ordenar lógicamente los emparejamientos para así poder contestar a preguntas tales como ¿Cómo de lejos está una demanda (u oferta) de una potencial contraparte? o ¿Cuáles son los requisitos que deberían ser dados o completados? [13]

Así, el Matchmaking semántico es definido como una tarea de Matchmaking donde las ofertas y demandas están expresadas con referencia a una especificación compartida de una conceptualización del conocimiento del dominio(como por ejemplo, una ontología) y donde los emparejamientos son realizados semánticamente mediante esa representación de conocimiento [12].

Los primeros sistemas de Matchmaking semántico nacen con la prometedora iniciativa de la Web Semántica, con el objetivo de operar en mercados electrónicos o en el descubrimiento de servicios web.

La primera propuesta bajo este enfoque es realizada por Gonzales-Castillo, Trastour y Bartolini que consideran la satisfacibilidad de conceptos en Descripción Lógica para la realización de emparejamientos. Esta propuesta es la antesala de las propuestas siguientes puesto que es la precursora en el uso de un sistema lógico que cuenta con un lenguaje para modelar las descripciones y con inferencias para realizar consultas lógicas.

A continuación aparece el sistema LARKS que permite la especificación de conceptos mediante ontologías y realiza el proceso de Matchmaking mediante semántica gracias a la inclusión del razonamiento deductivo de la subsunción.

A partir de aquí comienzan a surgir diferentes sistemas empleando distintos lenguajes de descripción como DAML+OIL, DAML-S o Descripción Lógica y distintos razonadores semánticos como CLASSIC, FaCT o MaMaS-tng, destacando a este último por ser específicamente desarrollado para la tarea de Matchmaking, el cual será detallado en el Capítulo 3 [13].

Dentro de estos sistemas de Matchmaking semántico han ido apareciendo diversos sistemas entre los que se podría destacar Jango, PersonaLogic, Kasbah o Websphere. A partir de los cuales se ha definido una categorización, que la mayoría de ellos presenta, para definir el proceso de Matchmaking o de emparejamiento entre una oferta y demanda con respecto a una ontología común. Esta categorización presenta cinco clases de emparejamiento [12]:

- Match exacto: Un match exacto se produce cuando la oferta y la demanda son equivalentes, es decir el emparejamiento es perfecto. Esta equivalencia de una forma lógica se expresa como Oferta $\equiv_T$ Demanda que se puede descomponer en la equivalente Oferta $\subseteq_T$ Demanda y Demanda $\subseteq_T$ Oferta.

- Match completo: Un match completo se produce cuando la oferta es más específica que la demanda pero además la oferta satisface completamente la demanda, es decir, la oferta tiene al menos todas las características requeridas por la demanda pero no necesariamente viceversa, siendo el proceso de Matchmaking no simétrico. Este tipo Match se expresa de forma lógica como Oferta $\subseteq_T$ Demanda.

- Match complementario: Un match complementario se produce cuando la demanda es más específica que la oferta y además la demanda satisface completamente la oferta, es decir, es el complementario del Match completo. Este tipo de Match se expresa de forma lógica como Demanda $\subseteq_T$ Oferta.

- Match potencial: Un match potencial se produce cuando la oferta no satisface completamente la demanda pero la intersección de ambas si es satisfacible, es decir, oferta y demanda tienen características en común y no presentan conflictos entre ellas. Este tipo de Match se expresa de forma lógica a partir de las expresiones Oferta $\not\sqsubseteq_T$ Demanda y Demanda $\sqcap$ Oferta $\not\sqsubseteq_T \bot$.

- Match parcial: Un match parcial se produce cuando la intersección entra la oferta y demanda no es satisfacible, es decir, existen conflictos entre la oferta y demanda. Este tipo de Match se expresa de forma lógica como Demanda $\sqcap$ Oferta $\not\sqsubseteq_T \bot$.

Como conclusión a los sistemas de Matchmaking y basado en sus aproximaciones actuales de sistemas semánticos, se puede definir el Matchmaking como un proceso que va más allá de simplemente encontrar a una oferta que coincida perfectamente con una demanda (o viceversa), si no que incluye descubrir todas las ofertas que hasta cierto punto puedan cumplir con la demanda (o viceversa) ofreciendo una clasificación y explicación para los emparejamientos [13].

Pero, por otro lado la tarea del Matchmaking sigue teniendo retos abiertos como la aparición de falsos positivos y negativos [9] o las descripciones de las ofertas y demandas que en ciertos dominios resultan heterogéneas, siendo difíciles de emparejar [11]. Ante estos retos se continúa con la investigación en esta área para conseguir un sistema óptimo que realice los emparejamientos más efectivos.

2.2 Razonadores semánticos

Como se ha indicado en el apartado anterior algunos de los sistemas de Matchmaking semántico presentados cuentan con un razonador semántico como parte del sistema para su correcto funcionamiento [12].

En general un razonador semántico es un sistema software que es capaz de inferir consecuencias lógicas a partir de un conjunto de aserciones o axiomas. Estos sistemas normalmente emplean lógica de primer orden para realizar las inferencias mediante el encadenamiento hacia atrás y hacia adelante.

En concreto este apartado hace especial hincapié en los razonadores de Lógica Descriptiva (Ver sección 3.3.4 para más información sobre Lógica Descriptiva) que son los empleados por los sistemas de Matchmaking citados en el apartado anterior. Los razonadores de Lógica Descriptiva son razonadores semánticos específicos para lenguajes de Lógica Descriptiva que algunas veces se encuentran integrados en sistemas mayores o simplemente son el sistema en sí. Como razonadores semánticos que son, el objetivo de estos es inferir consecuencias lógicas teniendo como principales problemas o inferencias la satisfacibilidad y la subsunción.

La satisfacibilidad de conceptos es una inferencia básica en estos sistemas ya que muchas otras inferencias se pueden reducir a la aplicación de la satisfacibilidad o no satisfacibilidad de los conceptos. Un concepto C es satisfacible con respecto a una ontología T, si la definición o expresión del concepto no denota al concepto vacío, de una manera más formal, se define como comprobar si existe un modelo I de T tal que C^I no es vacío [16].

La subsunción es una inferencia clave en estos sistemas de Descripción Lógica. La subsunción comprueba si un concepto D es considerado más general que uno denotado por C, de acuerdo a una clasificación o taxonomía de conceptos. La subsunción es una inferencia derivada de la satisfacibilidad ya que comprueba que en cada modelo I de T el conjunto denotado por C es un subconjunto del conjunto denotado por D [16].

Los razonadores de Lógica Descriptiva y en general los razonadores semánticos se caracterizan de acuerdo una serie de propiedades o características que presentan y que son fruto de las decisiones de diseño tomadas para su implementación. Las características más notables son [17]:

- Expresividad del lenguaje que soporta. Que se refiere al sub-lenguaje de Lógica Descriptiva que soporta para la representación del conocimiento.

- Algoritmo de razonamiento. Que se refiere al algoritmo de razonamiento lógico sobre el cual realiza las inferencias.

- Servicios de razonamiento. Que se refiere a los servicios o inferencias que provee para hacer razonamientos sobre una base de conocimiento.

- Tiempo de ejecución. Que se refiere al intervalo de tiempo que toma un servicio de inferencia en dar respuesta a lo solicitado.

- Soporte a interfaces externas. Que se refiere a las maneras que ofrece para acceder desde programas o aplicaciones externas a sus servicios de razonamiento.

De acuerdo a estas características en el Anexo I se muestra una tabla comparativa de los principales razonadores de Lógica Descriptiva que se han examinado, estos son FaCT++, HermiT, Pellet, RacerPro y MaMaS-tng.

Un breve análisis de esta comparativa ha mostrado las siguientes relaciones:

La expresividad del lenguaje de Lógica Descriptiva está ampliamente relacionada con el tiempo de ejecución de su algoritmo de razonamiento, sin embargo no está ligado a las inferencias que puede ofrecer ese razonador. Por tanto, como conclusión se obtiene que a la hora de emplear un razonador u otro dentro de un sistema de conocimiento uno de los principales aspectos a tratar es la expresividad de lenguaje que se requiere.

2.3 Ingeniería Ontológica

2.3.1 Introducción

El termino ontología proviene del griego teniendo su origen en la filosofía donde es una rama de la metafísica que estudia la naturaleza y estructura de la realidad, es decir, de lo que existe o es. Sin embargo, hoy en día este término ha derivado en dos enfoques o ramas distintas, por la una lado la rama filosófica de la ontología en la cual tiene su origen y por otro lado a la denominada rama informática o computacional [18].

Esta última rama es la más actual hoy en día y más relacionada con este proyecto. En la última década la palabra ontología ha cobrado una mayor importancia dentro de la Ingeniería del conocimiento. Una de las definiciones más aceptadas del término ontología dentro de esta rama es la que define a una ontología como una especificación explicita y formal de una conceptualización compartida [19]. Entendiendo, como conceptualización a un modelo abstracto de algún fenómeno en el mundo mediante el cual se identifican los conceptos relevantes de este fenómeno; como explicita a la definición explicita de los conceptos y limitaciones empleadas; como formal al hecho de que una ontología debe ser entendible por una máquina y como compartida a que captura conocimiento consensuado y aceptado por un grupo [20].

Otra definición de ontología es la de un vocabulario que representa conocimiento mediante la definición de conceptos y relaciones que describen y representan un área de interés. Estos vocabularios clasifican los términos, caracterizan las relaciones y definen posibles restricciones sobre ellos. Esta representación del conocimiento en vocabularios, dependiendo de su aplicación, puede ser utilizada en integración de datos, organización del conocimiento o "linked data", como terminologías comunes, o como un procedimiento de razonamiento para la solución de problemas [21].

Dadas estas posibilidades que ofrecen, las ontologías son ampliamente utilizadas en diversos campos como Ingeniería del Conocimiento, Inteligencia Artificial, Bioinformática o campos emergentes como la Web Semántica.

Además del término ontología, dado el creciente uso de estas, ha surgido también la Ingeniería de Ontologías o Ingeniería Ontológica que se refiere al conjunto de actividades que concierne el proceso de desarrollo de la ontología, su ciclo de vida y los métodos, metodologías, herramientas y lenguajes necesarios para su construcción. Esta Ingeniería surge a mediados de los 90, cuando hasta entonces el desarrollo de ontologías era más bien un arte y no una actividad de ingeniería como es ahora. Como se puede extraer de lo que será presentado a continuación, una ontología es un artefacto potente y

útil pero por otro lado complejo, por lo que su proceso de desarrollo debe ser metodológico. De esta manera el desarrollo de una ontología se puede equiparar al desarrollo de un producto software, de ahí la necesidad de la Ingeniería Ontológica al igual que el software cuenta con la Ingeniería del Software [20].

2.3.2 Componentes y modelado de Ontologías

Una ontología se construye a partir de un conjunto de distintos formalismos que tienen una función de modelado y representación específica. Estos formalismos son los siguientes:

- Conceptos o clases: Son las entidades básicas o primarias de la ontología que representan a objetos que se quieren formalizar y sobre los que se quiere tratar.

- Atributos: Son propiedades, rasgos o características que un concepto posee.

- Relaciones: Representan la manera por la cual los conceptos o instancias están relacionados o enlazados entre sí.

- Instancias: Son representaciones concretas de un concepto.

- Axiomas: Son proposiciones aplicadas sobre relaciones de conceptos para modelar la teoría que describe la ontología.

- Funciones: Son un caso especial de relaciones en las cuales el elemento n de la relación es único para los n-1 elementos precedentes.

Dependiendo del uso de unos u otros formalismos por parte de la ontología esta es definida como de un tipo u otro, distinguiendo así a las ontologías como "Ontologías Ligeras" y "Ontologías Pesadas". Por un lado las ontologías ligeras cuentan con conceptos, atributos, taxonomías de conceptos, relaciones e instancias. Mientras que por otro lado, las ontologías pesadas añaden a las ontologías ligeras axiomas y funciones para clarificar el significado de los términos representados en la ontología.

Para el modelado de las ontologías existen diferentes técnicas de modelado que ofrecen distintos componentes, paradigmas y lenguajes de representación del conocimiento. Sin embargo existen importantes limitaciones por una lado entre los componentes (conceptos, roles, etc.) usados para construir la ontología, los paradigmas de representación de conocimiento que representan tales componentes y el lenguaje usado para su implementación, lo que provoca que las distintas técnicas de modelado

presenten una capacidad de representación de conocimiento distinta en cuanto al grado de formalidad y granularidad que pueden expresar.

Así, desde comienzos de los 90, cuando aparecen las primeras ontologías, hasta hoy en día han aparecido distintas técnicas y lenguajes. A continuación se presentan las técnicas de modelado más extendidas:

- Modelado de ontologías usando Marcos y Lógicas de Primer Orden.
  Gruber fue el primero que propuso modelar ontologías usando Marcos y Lógicas de primer orden, en 1993. Para ello empleó cinco tipos de elementos: clases, relaciones, funciones, axiomas formales e instancias.

    - Las clases representan conceptos que pueden ser abstractos o específicos y son organizados en taxonomías lo que permite aplicar mecanismos de herencia.
    - Las relaciones representan un tipo de asociación entre los conceptos del dominio. Las ontologías normalmente contienen relaciones binarias en donde el primer argumento es denominado dominio y el segundo rango. La relación binaria más usual es "Subclass-of" empleada para construir la taxonomía de clases. Además con este modelado las relaciones binarias son también empleadas para definir atributos de conceptos, en donde el dominio es el concepto a describir y el rango es un tipo de dato.
    - Las funciones son un caso especial de la relación en el cual el n-ésimo elemento de la relación es único para los n-1 elementos.
    - Los axiomas formales modelan sentencias que son siempre verdaderas y son útiles para inferir nuevo conocimiento.
    - Las instancias son usadas para representar elementos concretos de las clases.

- Modelado de ontologías usando Lógicas Descriptivas.
  Las Lógicas Descriptivas serán descritas en detalle en el siguiente capítulo pero a modo resumen las Lógicas Descriptivas se pueden definir como formalismos lógicos que definen una teoría lógica la cual se compone de dos partes, el TBox y el ABox, en el TBox se define la terminología mediante conceptos y roles o relaciones entre los conceptos y en el ABox se definen los individuos o instancias.

- Modelado de ontologías usando técnicas de Ingeniería del Software.
  El Lenguaje Unificado de Modelado (UML) ha sido propuesto y mostrado en algunos proyectos como una herramienta para modelar ontologías. Algunas de las ventajas que presenta UML en el modelado de ontologías son que es fácil de entender para gente no relacionada con la Inteligencia Artificial y que cuenta con representaciones graficas de los modelos. Los componentes para modelar una ontología mediante UML son:

  - Las clases que son representadas mediante cajas UML, donde el nombre identifica al nombre del concepto y los atributos UML a los atributos del concepto.
  - Las relaciones que son representadas mediante asociaciones UML.
  - Los axiomas que son representados mediante el Lenguaje de Restricción de Objetos.

- Modelado de ontologías usando tecnologías de bases de datos.
  La ultima técnica de modelado presentado es la del modelado de ontologías mediante bases de datos. Las soluciones propuestas hacia este enfoque se basan en emplear diagramas Entidad-Relación para el modelado de la ontología, contando con los siguientes elementos:
  - Las clases que son representadas mediante entidades Entidad-Relación.
  - Los atributos que son atributos Entidad-Relación de la entidad.
  - Las relaciones que son relaciones Entidad-Relación entre entidades.

2.3.3 Proceso y Metodologías de desarrollo de Ontologías

El proceso de desarrollo de ontologías fue identificado en 1997 en el framework METHONTOLOGY —metodología para la construcción de ontologías. Esta propuesta se basó en el estándar de desarrollo software IEEE. Este proceso se refiere a las actividades que son desarrolladas cuando se construyen ontologías. Así, se aconseja llevar a cabo las actividades enmarcadas en las tres categorías siguientes.

- Actividades de gestión de la Ontología. Estas son tareas para gestionar el desarrollo de la Ontología entre las que se incluyen la planificación de las tareas a realizar, el control de las tareas para asegurar que se cumplen los plazos establecidos para cada tarea y el control de calidad para asegurar que las ontologías resultantes tienen la calidad deseada.

- Actividades orientadas al desarrollo de Ontologías. Este grupo de actividades se puede considerar el bloque o categoría principal aunque no implica que las otras actividades

no sean importantes o puedan omitirse para logar una ontología satisfactoria. Este grupo de actividades se subdivide en tres grupos que son el pre-desarrollo, desarrollo y post-desarrollo y que lógicamente siguen un orden temporal. Comenzando por el pre-desarrollo se estudia el entorno donde se va a emplear la ontología y preguntas acerca de la viabilidad de construir la ontología. A continuación se pasa al desarrollo donde se especifican los elementos de la ontología, se conceptualizan y se formalizan e implementan en algún lenguaje de Ontologías. Finalmente en el post-desarrollo se produce el mantenimiento de la ontología corrigiendo posibles errores o actualizándola.

- Actividades de apoyo a la Ontología. Estas actividades se llevan a cabo al mismo tiempo que el proceso de desarrollo e incluyen las actividades de adquisición de conocimiento, la evaluación de lo producido, la integración en el caso de usar ontologías ya construidas, la fusión en el caso de elaborar la ontología a partir de la mezcla de otras, el alineamiento entre las distintas ontologías y finalmente la documentación y configuración de la ontología.

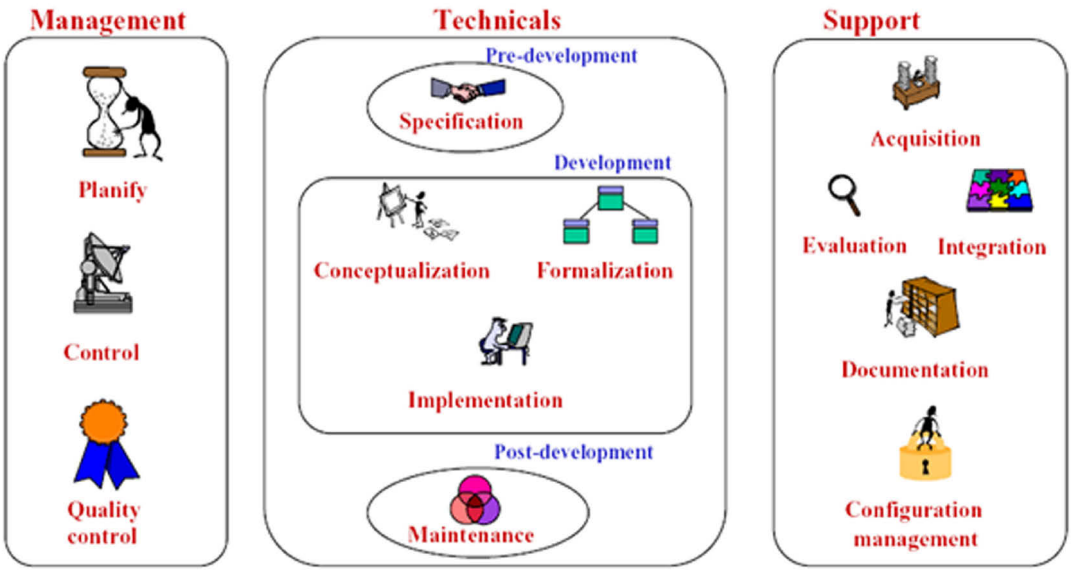


Figura 2-1: Proceso de desarrollo de ontologías

Desde la aparición de la Ingeniería Ontológica se han propuesto una serie de métodos y metodologías para el desarrollo de ontologías. Algunas de estas soluciones son llamadas métodos porque simplemente fueron los métodos que se siguieron en el desarrollo de una ontología concreta dentro de un proyecto como pueden ser "CycMethod" o "Uschold and King'sMethod" mientras que por otro lado aparecen metodologías elaboradas para el desarrollo de cualquier ontología como "Methontology" o "On-To-Knowledge". La mayoría de estos métodos y metodologías se centran en las actividades de desarrollo —especialmente en la conceptualización e implementación [20].

Capítulo 3

TECNOLOGÍAS EMPLEADAS

3.1 Tecnologías Web

Desde la explosión de Internet y por tanto de la Web o WWW, cada día cobran mayor relevancia las páginas o portales web que ofrecen distintos tipos de servicios y/o recursos. Por ello, desde el inicio del desarrollo de las primeras páginas web hasta el día de hoy, han aparecido distintas tecnologías o lenguajes con el objetivo de ser empleados en el desarrollo de estos, aportando nuevas herramientas para crear sitios más vistosos, potentes, etc. De esta manera, a día de hoy, para desarrollar un portal web que combine diferentes características y/o propiedades se necesita hacer uso de varias de estas tecnologías. Estas tecnologías se dividen en tecnologías del lado del cliente y del lado del servidor en función del entorno en donde se ejecutan. En la siguiente figura se muestran las distintas tecnologías:

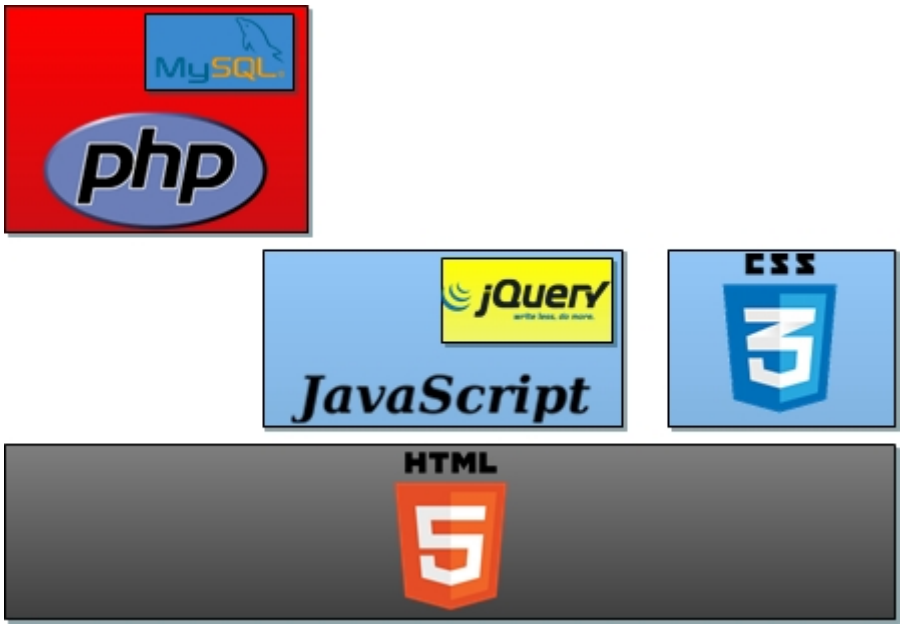


Figura 3-1: Pila de las principales tecnologías Web

Por otro lado, las páginas web creadas con estas herramientas y tecnologías se pueden dividir en dos tipos: estáticas y dinámicas.

Una página web estática es aquella que es presentada al usuario exactamente como esta almacenada, mostrando la misma información a todos los usuarios. Mientras que, una página web dinámica es generada por una aplicación web, presentando distinto contenido en función de ciertas variables.

3.1.1 HTML

HTML o "Lenguaje de Marcado de Hipertexto" es el lenguaje de programación que ejerce el papel de tecnología núcleo o referencia para la elaboración de páginas web. El código HTML se escribe en forma de etiquetas que marca de una manera específica lo que se escribe entre estas etiquetas, las cuales son interpretadas por los navegadores web para dar formato al texto, incluir enlaces a otros sitios o presentar elementos multimedia [22].

De esta manera el lenguaje HTML se encarga de mostrar información de una manera más o menos grafica pero de una manera inteligente por lo que para ofrecer mayores posibilidades se hace necesario el uso de otras tecnologías que complementen a esta [23].

3.1.2 CSS

CSS o "Hojas de estilo en cascada" es un lenguaje que permite describir cómo será mostrado o presentado un documento escrito en un lenguaje de marcado. Generalmente es empleado como un completo al HTML para dotar a este de un mecanismo que proporcione un estilo al documento, entendiendo por estilo al aspecto y formato de lo visualizado y que se basa en los colores, fuentes, tamaños, etc. de los elementos presentados en el documento HTML. Separando así, por un lado el contenido, generado por el HTML y por otro, la presentación de este contenido generada por el CSS [22].

El código CSS se basa en la declaración de reglas que definen el estilo de los elementos HTML. Estando compuesta una regla por dos partes: un selector que indica al tipo de elemento que se aplica y la declaración que establece las propiedades de esos elementos [24].

3.1.3 JavaScript

JavaScript [23] es un lenguaje de programación interpretado o de scripting que es ejecutado por el navegador web en el lado del cliente. El código JavaScript interactúa con el documento escrito en HTML mediante el DOM o "Modelo de Objetos del

Documento" para dotar al sitio web de mejoras en la interfaz de usuario o manejar información dinámica.

3.1.4 jQuery

jQuery [25] es una biblioteca de JavaScript que ofrece una serie de funcionalidades basadas en el uso de bibliotecas propias de este lenguaje. Es decir es una abstracción de código que permite hacer las mismas cosas que con JavaScript pero de una manera más rápida, simple y sencilla.

3.1.5 PHP

PHP [26] es un lenguaje de programación interpretado o de scripting que es ejecutado en el lado del servidor. Aunque puede ser usado para crear cualquier tipo de aplicación fue diseñado originalmente para la creación de páginas web dinámicas, siendo actualmente el lenguaje del lado de servidor más extendido en la Web. El código escrito en PHP es procesado por un intérprete que genera el contenido HTML de manera dinámica y lo envía al cliente.

Además, destaca por contar con diversos frameworks que permiten crear contenidos dinámicos con una relativa facilidad.

3.2 Redes Sociales

3.2.1 Elgg

Elgg [27] es un potente framework o motor de desarrollo para sitios web sociales, especialmente redes sociales, desarrollado en lenguaje PHP. Su desarrollo es gestionado por la fundación no lucrativa Elgg, pero el hecho de ser una plataforma Open Source hace que exista una gran comunidad de desarrolladores que contribuyen a hacer a Elgg mayor y mejor.

Elgg cuenta en su ADN con una red social que provee todas las funciones sociales más demandadas hoy día, como son:

Actividad: Que permite a los usuarios seguir todo lo que está ocurriendo en la red social.

Perfiles: Que describen las caracterícelas, gustos, etc. de los usuarios en la red social.

Grupos: Que permiten crear comunidades virtuales de usuarios en las que poder hablar y tratar sobre temas específicos.

Blogs: Que permiten a los usuarios crear contenidos acerca de diversos temas.

Microblogging: Que permite a los usuarios escribir sobre algún tema de manera breve y compartirlo en tiempo real con otros usuarios.

Notificaciones: Que permite a los usuarios estar actualizados con las últimas novedades que están ocurriendo en la red social.

Amigos: Que permite a los usuarios crear relaciones de amistad con otros usuarios de la red social.

Etiquetado: Que permite a los usuarios marcar como favorito contenidos presentes en la red social.

Elgg puede ser instalado en un servidor web y ser usado inmediatamente, sin requerir ningún tipo de desarrollo extra, contando con todas las características sociales anteriores. Sin embargo puede ser personalizado mediante la inclusión de plugins.

Los plugins son componentes claves en Elgg que permiten dotar de mayor funcionalidad al núcleo de funcionalidades y características de Elgg. Siendo así, Elgg altamente extensible, ofreciendo como parte de su núcleo herramientas y/o bloques de construcción para crear plugins que sean una parte más del sistema y estén perfectamente integrados en este.

Elgg sigue el patrón de arquitectura software Modelo Vista Controlador (MVC) el cual separa los datos de las interfaces y de los eventos o comunicaciones que es implementado de la siguiente manera:

- Modelo de datos: En Elgg todos los datos siguen un modelo de datos unificado basado en unidades atómicas de datos que son denominadas entidades. Las entidades son una abstracción de la información contenida en la base de datos interna de Elgg. Con el uso de estas entidades por parte de los plugins se evita que se tengan que desarrollar bases de datos externas favoreciendo a que el sistema sea más estable y que los contenidos de los distintos plugins puedan ser mezclados de una manera consistente.
  Todas las entidades en el sistema tienen al menos como padre a la clase ElggEntity de la cual heredan cuatro clases de entidades: ElggObject, ElggUser, ElggSite y ElggGroup. De entre estos cuatro tipo de entidades destaca la clase ElggObject, la cual permite representar cualquier tipo de dato externo al modelo

de Elgg mediante el uso de dos clases auxiliares: ElggMetadata y ElggAnnotation.

- Vistas: Para separar los datos de la forma en la que son presentados, Elgg cuenta con un sistema de presentación de los datos mediante plantillas. Estas plantillas son denominadas vistas y se encargan de generar lo que se ve, es decir, generan el código HTML que es enviado al navegador para ser mostrado. Estas vistas manejan tanto el diseño de las páginas como la generación de los elementos visuales que componen las páginas. Cada vista se encarga de crear una sección de presentación a partir de una entrada de datos y al igual que las entidades hay distintos tipos de vistas. Además cada tipo de vista suele asociarse a un tipo de entidad.

- Sistema de eventos: Elgg tiene un sistema de eventos que puede ser usado para añadir o reemplazar funcionalidad al núcleo. Estos eventos son activados para realizar acciones cuando ocurren ciertas cosas para las cuales han sido registrados. Para la creación de estos eventos se dispone de dos tipos de clases: Elgg events y plugin Hooks.

3.3 Web Semántica

3.3.1 OWL

OWL [28] o "Web Ontology Language" es un lenguaje de marcado semántico para la publicación e intercambio de información en la Web. El objetivo de OWL es representar el significado de los términos y las relaciones de términos contenidos en ontologías o vocabularios de manera que la información pueda ser entendida por máquinas y aplicaciones. OWL está construido sobre RDF y codificado en XML y juega un importante rol en la construcción de la Web Semántica. Actualmente la definición del lenguaje OWL se encuentra por su segunda versión, pero desde su primera versión se ofrecen tres sub-lenguajes los cuales ofrecen un distinto grado de expresividad para ser usados en entornos distintos.

3.3.2 RDF

RDF [29] o "Resource Description Framework" es un lenguaje para representar información acerca de recursos en la Web, más concretamente, para representar metadatos de los recursos. Tales representaciones permiten que la información representada pueda ser procesada por máquinas y aplicaciones y no sirva solo para ser mostrada a humanos, favoreciendo así al intercambio de datos en la Web.

RDF se basa en representar recursos describiéndolos mediante propiedades y valores a esas propiedades. Cada descripción del recurso es lo que se denomina Tripleta o Grafo (en su representación gráfica) y se compone de tres elementos: sujeto, que es el recurso a describir; predicado, que define la propiedad; y objeto que es el valor asignado a la propiedad. Además, para identificar a cada uno de estos tres elementos se hace uso de identificadores Web, URIs.

3.3.3 SPARQL

SPARQL [30] o "Protocol and RDF Query Language" es un lenguaje de consulta de grafos RDF, es decir, un lenguaje para consultar, recuperar y manipular información almacenada en formato RDF.

Al igual que RDF, las consultas SPARQL contienen conjuntos de tripletas, pero en este caso los elementos (sujeto, predicado y objeto) de las tripletas pueden ser una variable. SPARQL cuenta con cuatro tipos de consultas posibles: consultas de tipo SELECT, consultas de tipo CONSTRUCT, consultas de tipo ASK y consultas de tipo DESCRIBE.

SPARQL únicamente es el lenguaje de consulta, sin embargo existen diversos entornos SPARQL que además del lenguaje de consulta incluyen motores para el mantenimiento de los datos.

- Apache Jena
  Entre esas aplicaciones se encuentra Jena [31], que es un framework Java para crear aplicaciones de Web Semántica. Jena proporciona un conjunto de herramientas y bibliotecas Java que ayudan en el desarrollo de este tipo de aplicaciones, entre estas herramientas se encuentra un motor de consulta SPARQL.

3.3.4 Lógicas Descriptivas

Las Lógicas Descriptivas [16] son una familia de formalismos lógicos para la representación del conocimiento. Son lógicas más expresivas que la lógica proposicional y además cuentan con servicios de razonamiento. Actualmente tienen una gran importancia en la representación de Ontologías y en la Web Semántica.

Las Lógicas Descriptivas son útiles para el diseño de sistemas basados en conocimiento, en los cuales no solo se busca contar con un lenguaje para definir una base de conocimiento sino que también se requieren de servicios de razonamiento que permitan realizar inferencias sobre la base de conocimiento. Así, por un lado el lenguaje permite representar el conocimiento de un dominio de aplicación dado de una forma

estructurada y formal y por otro lado proveen servicios de razonamiento que basan su funcionamiento en verificar consecuencias lógicas.

Todas las Lógicas Descriptivas están dotadas de una sintaxis y semántica específica, sin embargo todas ellas tienen elementos básicos que son:

- Conceptos: Que son usados para describir conjuntos de objetos reales. Por ejemplo, "Persona" sería un concepto.

- Roles: Que son usados para enlazar o relacionar conceptos. Por ejemplo, "tieneHijo" sería un rol que enlazaría a dos personas.

- Individuos: Que representan elementos específicos pertenecientes a un concepto. Por ejemplo, "María" sería un individuo de Persona.

Estos elementos básicos pueden ser combinados mediante constructores para formar expresiones de conceptos y roles que describen e identifican conceptos más complejos. Así, es posible combinar conceptos mediante constructores como disyunción($\cup$), conjunción($\cap$) y negación atómica($\neg$) o identificar relaciones mediante constructores como cuantificación existencial($\exists$), cuantificación universal($\forall$) o restricciones numéricas($\leq$, $\geq$) y otros constructores como el concepto universal($\top$) y el concepto vacío($\bot$).

Así por ejemplo podemos tener conceptos complejos como: "Estudiante $\cap$ $\neg$Femenino" que describe a todos los estudiantes que no son chicas o "Estudiante$\cap$ $\exists$tienePadre.Profesor" que describe a todos los estudiantes que al menos tienen un padre que es profesor.

Dependiendo de los constructores que tenga el Lenguaje de Descripción Lógica se diferencian distintos lenguajes o sub-lenguajes que son nombrados siguiendo una nomenclatura informal, que describe los operadores que ofrece, codificando cada uno de estos con una etiqueta añadida a la de la lógica de partida que contiene los elementos básicos, siendo esta nomenclatura la siguiente:

AL [U] [E] [N] [C]

Dónde:

- AL se refiere al lenguaje base que cuenta con conceptos atómicos, concepto universal, concepto vacío, negación atómica, intersección, cuantificador universal y cuantificador existencial limitado al concepto universal.
- U se refiere al lenguaje que además cuenta con la disyunción.

- E se refiere al lenguaje que además cuenta con el cuantificador existencial completo.
- N se refiere al lenguaje que además cuenta con restricciones numéricas mediante las restricciones at-least($\geq$) y at-most($\leq$).
- C se refiere al lenguaje que además cuenta con las negaciones de conceptos arbitrarios.

Los sistemas de Lógica Descriptiva proveen las funcionalidades para construir la base de conocimiento e inferencias para realizar consultas a la base de conocimiento. Estas funcionalidades son ofrecidas a través una interfaz conocida como "Tell&Ask", la cual especifica operaciones que permiten la construcción de la base de conocimiento (Operaciones Tell) y operaciones que permiten obtener información de la base de conocimiento (Operaciones Ask).

La base de conocimiento de estos sistemas se divide en dos componentes denominados: TBox y ABox. El TBox contiene declaraciones que definen y describen toda la terminología de la base de conocimiento es decir los conceptos y roles. Las declaraciones son conjuntos de axiomas construidos mediante la combinación de conceptos con los operadores de definición ($\equiv$) e inclusión ($\subseteq$). El operador de definición define un nuevo concepto en términos de otros conceptos previamente definidos y puede ser interpretado como una relación de equivalencia. El operador de inclusión jerarquiza los conceptos en forma de taxonomía mediante la relación de subsunción, la cual relaciona dos conceptos de los cuales uno está incluido dentro de otro.

Por ejemplo una definición podría ser "Mujer$\equiv$ Persona $\cap$ Femenino" que define a una mujer como una persona del sexo femenino y una inclusión podría ser "SerHumano$\subseteq$ Persona" que establece que persona es un ser humano.

El ABox contiene aserciones acerca del dominio es decir instancias definidas mediante conceptos y roles definidos en el TBox.

Por otro lado, como se ha comentado, los sistemas de Lógica Descriptiva proporcionan servicios de razonamiento mediante inferencias lógicas. Estas inferencias lógicas son: La satisfacibilidad y la subsunción. La satisfacibilidad comprueba si existe una interpretación lógica no vacía para un concepto dado. La subsunción comprueba si un concepto es más general que otro.

3.3.5 DIG

DIG [32] o también denominada Interfaz DIG para Lógica Descriptiva es una interfaz para el acceso uniforme a razonadores de Lógica Descriptiva. Ante el creciente uso de razonadores de Lógica Descriptiva nace la necesidad de contar con una interfaz común y estándar que permita, por una lado, a los clientes interactuar con diferentes razonadores de una manera uniforme y por otro lado, a los desarrolladores de los razonadores no tener que concentrarse en ofrecer un abanico de servicios para acceder al razonador.

Esta interfaz define un protocolo de comunicación con los razonadores de Lógica Descriptiva basado en el protocolo HTTP para enviar mensajes entre razonador y cliente. Los mensajes siguen un esquema XML, definido por la interfaz DIG, que describe el lenguaje conceptual y operaciones complementarias.

Cada interacción con un razonador DIG es iniciada por un cliente DIG que realiza peticiones HTTP POST codificadas según el esquema XML DIG al razonador. A continuación el razonador contesta con un código de respuesta HTTP y si es necesario los resultados a la operación codificados según el esquema DIG.

El esquema DIG provee tres clases de acciones para ser usadas por el cliente que son: operaciones de gestión, operaciones Tell y operaciones Ask.

Las operaciones de gestión contienen las peticiones para solicitar una nueva base de conocimiento al razonador y para eliminar una base de conocimiento creada.

Las operaciones Tell contienen peticiones para definir la base de conocimiento, es decir operaciones para crear declaraciones de conceptos, roles o instancias dentro de la base de conocimiento.

Las operaciones Ask contienen peticiones de consulta a la base de conocimiento.

Además de estas operaciones, el esquema DIG define todos los elementos que componen el lenguaje soportado y que es usado en las peticiones anteriores y también codifica las correspondientes respuestas a cada tipo de petición.

- DIG 1.1 XMLBeans API
  DIG 1.1 XMLBeans API [33] es una API Java para el análisis, validación, creación y manipulación de documentos XML conforme al esquema XML DIG 1.1 desde aplicaciones Java.

3.3.6 MaMaS-tng

MaMaS-tng [12] [13] [15] o "Match Making Service – The Next Generation" es un motor de razonamiento de Lógica Descriptiva que cuenta con un sub-lenguaje ALN. MaMaS-tng ofrece todos los servicios clásicos de inferencia estándar pero también ofrece servicios de Matchmaking mediante servicios de inferencia no estándar, que permiten categorizar y clasificar los matches de acuerdo a su semántica. MaMaS-tng ha sido desarrollado por el Information System Laboratory de la Universidad Politécnica de Bari.

MaMaS-tng es un sistema multiusuario y multiontología basado en servlets Java que está disponible como un servicio HTTP a través de un interfaz compatible con DIG 1.1. Esta especificación DIG 1.1 ha sido modificada y adaptada para poder tratar los servicios de inferencia no estándar que ofrece MaMaS-tng y no otros razonadores lógicos, esta especificación será descrita brevemente en este apartado.

El sistema está fuertemente relacionado con el sistema CLASSIC ya que incrusta una versión modificada del razonador NeoClassic, que es una implementación de éste. Esta modificación afecta al algoritmo estructural de Subsunción de CLASSIC para así permitir la categorización y clasificación de matches. Aunque el sistema podría funcionar con otros razonadores como FaCT o Racer, la elección de CLASSIC se debe a que sus inferencias toman un tiempo Polinomial que permite realizar operaciones de una manera síncrona incluso con grande ontologías.

- Servicios soportados

MaMaS-tng al igual que cualquier motor de razonamiento de Lógica Descriptiva ofrece servicios de inferencia estándar como subsunción, satisfacibilidad y algunos otros más derivados de estos dos, sin embargo, también cuenta con dos inferencias no estándar, la Abducción y Contracción gracias a las cuales ofrece las funcionalidades de Matchmaking semántico de categorización y clasificación. Estos servicios ofrecen una explicación mayor a la hora de realizar matches no quedándose en solamente dar básicas respuestas booleanas, lo que permite crear un espacio de negociación entre demanda y oferta.

Así, MaMaS-tng permite alojar Bases de Conocimiento en forma de ontologías, que son identificadas mediante una URI. Y sobre estas bases de conocimiento es posible realizar múltiples inferencias entre las que destacan inferencias específicas de MaMaS-tng que son:

- Detección del Tipo de Match: La detección del tipo de Match permite determinar la categoría de un Match entre una demanda y oferta dadas y definidas a partir de

una ontología. Teniendo entre los posibles tipos de Match: Exacto, Completo, Complementario, Potencial y Parcial.

- Ranking Potencial: El ranking potencial calcula una medida numérica que expresa la distancia entre una demanda y oferta dadas y definidas a partir de una ontología. Esta distancia expresa el número de características explicitadas en la demanda que no son cumplidas por la oferta.

- Ranking Parcial: El ranking parcial calcula una medida numérica que expresa la distancia entre una demanda y oferta dadas y definidas a partir de una ontología. Esta distancia expresa el número de características que están en conflicto entre la demanda y la oferta.

- Abducción: La Abducción se aplica sobre un Match de tipo potencial formado por una oferta y una demanda y determina una hipótesis acerca de qué características no están especificadas en la oferta de cara a satisfacer completamente la demanda.

- Contracción: La Contracción se aplica sobre un Match de tipo parcial formada por una oferta y una demanda y determina una explicación de porqué la oferta y la demanda no son compatibles y presentan conflictos.

- Fundamentos teóricos de MaMaS-tng

En este aparatado se presentan algunos fundamentos teóricos en los que se basa MaMaS-tng y características teóricas que presenta.

La base del funcionamiento de MaMaS-tng se encuentra en la implementación de algoritmos para ofrecer la clasificación entre ofertas y demandas. Primeramente hay que definir correctamente los enfoques de oferta y demanda que tiene MaMaS-tng ya que no se tienen porque corresponder al 100% con sus tradicionales significados. MaMaS-tng ofrece una clasificación que puede ser tanto de demandas con respecto a ofertas como al contrario, de ofertas con respecto a demandas, entendiendo a estos términos con sus tradicionales significados. Por ello, MaMaS-tng interpreta como demanda al elemento usado para realizar la búsqueda que puede ser tanto una oferta como una demanda y oferta al elemento que actúa como contraoferta o contraparte y puede ser tanto una oferta como una demanda. Con esta visión se tiene la siguiente notación para comprender sus principales definiciones.

La “L” se refiere una Lógica Descriptiva genérica, la “T” a un TBox que representa una ontología común definida conforme L, la “S” una oferta bajo la interpretación anterior y

satisfacible en T y la "D" una demanda bajo la interpretación anterior y satisfacible en T.

- Definición 1: MaMaS-tng interpreta descripciones de Mundo Abierto. Esta definición se refiere a que la ausencia de una característica en la descripción de una "S" o "D" no es tomada como una limitación, permitiendo así especificaciones incompletas. Esto supone que características o detalles que no hayan sido descritos pueden ser refinados después o dejarlos abiertos porque sean irrelevantes para el usuario.

- Definición 2: MaMaS-tng implementa una función de penalización no simétrica. Esta definición se refiere a que la aplicación de una función de penalización o clasificación sobre una demanda y oferta no devolverá el mismo resultado dependiendo desde los dos posibles punto de vista de aplicaciones decir, en términos matemáticos $p(D,S,T) \neq p(S,D,T)$.

- Definición 3: MaMaS-tng presenta una independencia sintáctica en sus funciones de clasificación. Esta definición se refiere a que tanto en su función de clasificación parcial como potencial para un par de ofertas S_1 y S_2 y una demanda D, S_1 y S_2 tendrán la misma clasificación con respecto a D, siendo S1 y S2 lógicamente equivalentes independientemente de su descripción sintáctica.

- Definición 4: MaMaS-tng implementa una función de penalización para clasificación potencial monótona sobre la subsunción. Esta definición se refiere a que para un par de ofertas S_1 y S_2, siendo $S_1 \subseteq S_2$, y una demanda D, la aplicación de una clasificación potencial resultará siempre en que S_2 será clasificada mejor o igual que S_1.

- Definición 5: MaMaS-tng implementa una función de penalización para clasificación parcial no monótona sobre la implicación. Esta definición se refiere a que para un par de ofertas S_1 y S_2, siendo $S_2 \subseteq S_1$, y una demanda D, la aplicación de una clasificación potencial resultará siempre en que S_2 será clasificada peor o igual que S_1.

A partir de estas definiciones teóricas están construidos sus 4 algoritmos principales los cuales son:

- Contracción: La contracción se presenta cuando la conjunción de D y S es insatisfacible en T, es decir, existen contracciones o conflictos entre demanda y oferta. El objetivo de la contracción es por un lado retraer requisitos en D para obtener un concepto K de requisitos a mantener de manera que $K \cap S$ sea

satisfacible, es decir mantener los requisitos que no están en conflicto y por otro lado conocer los requisitos que están en conflicto y que deberán ser negociados. Esto último supone obtener un concepto G de conceptos en conflicto de manera que $G \cap K = D$. Así la contracción es formulada por MaMaS-tng como una función CCP(Problema de contracción de conceptos) con entrada {L,D,S,T} y salida {G,K}.

- Abducción: La abducción se presenta cuando la conjunción de D y S es satisfacible en T, pero la oferta no implica la demanda, es decir existen diferencias entre demanda y oferta. El objetivo de la abducción es formular una hipótesis de características que deben ser dadas para que D y S sean equivalentes. Esto supone obtener un concepto H tal que $H \subseteq D$ y $S \cap H$ es satisfacible. Así la abducción es formulada por MaMaS-tng como una función CAP(Problema de abducción de conceptos) con entrada {L,S,D,T} y salida {H}.

- Ranking potencial: El Ranking potencial se presenta cuando se dan una oferta y demanda que no presentan contradicciones entre sí, pero no son equivalentes. MaMaS-tng formula el ranking potencial como una función denominada rankPotential que toma como entrada {S,D,T} y devuelve como salida un numero entero positivo que representa la similitud entre la demanda y oferta. El algoritmo es implementado a partir de la abducción de conceptos computando el número de conceptos que contiene el concepto abducido H.

- Ranking parcial: El Ranking parcial se presenta cuando se dan una oferta y demanda que presentan contracciones entre sí. MaMaS-tng formula el ranking parcial como una función denominada rankPartial que toma como entrada {S,D,T} y devuelve como salida un numero entero positivo que representa el número de contradicciones entre la demanda y oferta. El algoritmo es implementado a partir de la contracción de conceptos computando el número de conceptos que contiene el concepto contraído G.

- Interfaz DIG 1.1 extendida.

Como se comentó al principio de este apartado, MaMaS-tng presenta una interfaz DIG mediante la cual ofrece sus servicios a terceras aplicaciones. De esta manera como ante cualquier razonador con una interfaz DIG, es posible acceder a sus servicios mediante peticiones HTTP a su URL, en este caso a http://dee227.poliba.it:8080/MAMAS-tng/DIG.

Esta interfaz DIG es la versión 1.1, pero ha sido extendida [34] para poder ofrecer los servicios específicos que presenta MaMaS-tng como son el rankPotential o el

rankPartial entre otros. De esta manera las operaciones de esta versión extendida se estructuran según las 3 clases de acciones de DIG pero algunas de ellas presentan alguna modificación y también existen nuevas.

Dentro de las operaciones de gestión cabe destacar la petición <newKB> que permite la creación de una nueva base de conocimiento, que ha sido modificada para poder indicar el tiempo de mantenimiento de la base de conocimiento en el razonador y la propiedad de compartición de la base de conocimiento.

Dentro de las operaciones Tell solo aparece una nueva petición, la petición <clearindividual> que permite eliminar un individuo de la base de conocimiento.

Y finalmente, dentro de las operaciones Ask es donde se encuentran las principales operaciones añadidas para dar soporte a las funcionalidades específicas de MaMaS-tng, estas son: <abduce> para realizar la abducción de conceptos, <contract> para realizar la contracción de conceptos, <matchType> para conocer el tipo de Match entre una demanda y oferta y <rank> para realizar un ranking entre una demanda y oferta.

3.4 Java

Java [35] es un lenguaje de programación de propósito general orientado a objetos, que surgió en 1995, desarrollado por James Gosling de Sun Microsystems. El propósito inicial del lenguaje era ser usado por aplicaciones empotradas en dispositivos electrónicos como electrodomésticos. El diseño del lenguaje está influenciado por otros lenguajes de programación como Smalltalk o C y C++ de los que toma elementos sintácticos, sin embargo elimina herramientas de bajo nivel de estos, como la gestión de memoria. Actualmente es definido por Sun como un lenguaje de alto nivel, simple, distribuido, multihilo, dinámico, portable, robusto, seguro y de alto rendimiento.

El lenguaje Java se encuentra ligado a la plataforma Java, la cual se compone de la Máquina Virtual de Java y la API Java.

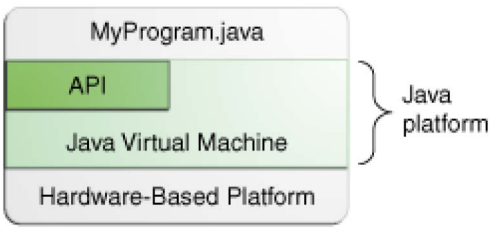


Figura 3-2: Esquema del entorno de ejecución de la plataforma Java

La Máquina Virtual de Java se encarga de ejecutar los programas escritos en Java y compilados. Estos programas son compilados a bytecode que es el lenguaje máquina de la Máquina Virtual de Java, en lugar de ser compilados en el código maquina nativo del procesador que lo ejecuta. Esto permite que un programa Java pueda ser ejecutado en cualquier plataforma sin tener que ser reescrito ni recompilado. Por otro lado la API Java, es una colección de componentes que ofrece diversas funcionalidades, a través de librerías denominadas paquetes, para el desarrollo de aplicaciones en lenguaje Java.

Gracias a las características que lo definen, el lenguaje Java ha tenido un gran apoyo por parte de la industria informática lo que ha derivado en un crecimiento constante desde sus inicios, haciéndolo uno de los lenguajes más usados y exitosos a día de hoy [36]. Este amplio uso hace que exista una gran cantidad de librerías que suponen un ahorro de tiempo en el desarrollo de software nuevo.

3.5 MySQL

MySQL [37] es un sistema de gestión de bases de datos relacionales (RDBMS) propiedad de Oracle. MySQL es uno de los sistemas de gestión de bases de datos más usados mundialmente gracias a algunas de sus propiedades más relevantes como ser software open source, multihilo y multiusuario.

MySQL es ampliamente utilizado en aplicaciones Web como componente de la pila LAMP o WAMP, pero además es también utilizada para la gestión de bases de datos de aplicaciones de escritorio ya que existen API's específicas para diversos lenguajes que incluyen librearías para el acceso a estas bases de datos.

- JDBC

JDBC [38] o "Java Database Connectivity" es una API Java que permite la conexión de cualquier aplicación Java con la API nativa específica de un sistema gestor de bases de datos relacionales. Esta API presenta una colección de métodos para la gestión de manejadores de conexión de la base de datos que permiten la ejecución de operaciones sobre la base de datos utilizando el lenguaje SQL.

3.6 Procesamiento de Lenguaje Natural

3.6.1 OpenNLP

OpenNLP [39] es una librería Java desarrollada por Apache para el procesamiento de texto en Lenguaje Natural. Por medio de su API ofrece herramientas para realizar las tareas más comunes en Procesamiento de Lenguaje Natural como son: Tokenización, detección de oraciones, etiquetado morfológico, detección de entidades, análisis sintáctico y análisis semántico.

3.6.2 WordNet

WordNet [40] es una base de datos léxica de términos en inglés que almacena nombres, verbos, adjetivos y adverbios. Todas las palabras almacenadas son agrupadas en conjuntos de sinónimos denominados synsets, los cuales expresan un concepto distinto cada uno. Estos synsets están interconectados mediante relaciones semánticas y sintácticas atribuidas a los conceptos que expresan, de esta manera WordNet no es solo un diccionario de sinónimos sino que es una potente red semántica de palabras y conceptos. Esta red semántica contiene relaciones entre los conceptos de sinonimia, hiperonimia, hiponimia, meronimia, troponimia y antonimia.

- JAWS
  WordNet puede ser consultado directamente de manera online mediante su interfaz web, sin embargo existen múltiples interfaces que permiten su consulta desde programas externos desarrollados en diversos lenguajes. Una de estas interfaces es JAWS [41] o "Java API for WordNet Searching" que como su nombre expresa es una es una API Java simple y rápida que proporciona herramientas y librerías para realizar consultas a la base de datos de WordNet.

3.6.3 Google Spell Checker

Google Spell Checker [42] es un software de corrección ortográfica de Google. Este software es usado por Google en las consultas de búsqueda realizadas a su buscador de donde resulta la conocida notificación "Quizás quisiste decir" que es mostrada cuando las palabras empleadas para realizar la búsqueda contienen errores lexicográficos.

- Google API Spelling Java
  Google API Spelling Java [43] es una API Java que permite la corrección ortográfica desde una aplicación Java haciendo uso del servicio Google Spell Checker. Esta API basa su funcionamiento en realizar peticiones web al servicio.

Capítulo 4

ANÁLISIS DEL SISTEMA

4.1 Especificación de Requisitos

4.1.1 Introducción

En este apartado se presenta la Especificación de Requisitos Software (ERS) para el sistema "AFRICA BUILD Portal - Mobility Brokerage Service" propuesto para ser desarrollado en este Trabajo Fin de Grado. La estructura y formato de este apartado sigue el formato y directrices recomendadas por el estándar "IEEE Recommender Practice for Software Requirements Specifications" (ANSI/IEEE Std. 830, 1998).

4.1.1.1 Propósito

El objetivo de esta especificación es presentar de forma clara y precisa una lista de funcionalidades y restricciones en términos de requisitos software que presentará el sistema a construir. Esta lista de requisitos además de servir para especificar las funcionalidades y restricciones del sistema también servirá para verificar y validar la corrección e integridad del mismo.

Esta especificación de requisitos está dirigida, por un lado a futuros desarrolladores que puedan necesitar conocer el sistema para una posible modificación y por otro lado, a potenciales usuarios que quieran conocer que les puede ofrecer el sistema.

4.1.1.2 Ámbito del sistema

El sistema "AFRICA BUILD Portal - Mobility Brokerage Service" se enmarca en el portal social AFRICA BUILD Portal como un componente más de este y que tratará de ser una herramienta de soporte para la gestión de ofertas y demandas de movilidad a través del portal social.

El sistema servirá para la publicación de ofertas y demandas de movilidad por parte de los usuarios de AFRICA BUILD Portal, pero su principal ayuda estará en la generación de recomendaciones. El concepto de recomendación se refiere a la sugerencia por parte del sistema a los usuarios de posibles ofertas y/o demandas de movilidad que se asemejen a sus demandas y ofertas.

4.1.1.3 Definiciones, Acrónimos y Abreviaturas

En esta sección se presentan una serie de definiciones, acrónimos y abreviaturas que serán utilizados en el resto de esta especificación de requisitos y que sus significados pueden no estar claros.

- Definiciones:

Termino	Definición
AFRICA BUILD	AFRICA BUILD es una acción coordinada entre distintas instituciones con el objetivo de apoyar y desarrollar avanzados centros de excelencia en la atención a la salud, educación e investigación en países Africanos.
AFRICA BUILD Portal	AFRICA BUILD Portal es una red socio-profesional con el objetivo de ofrecer la posibilidad de crear comunidades virtuales para fomentar la investigación y educación en países Africanos.
Demanda	Una demanda es una solicitud presentada por una persona con el objetivo de encontrar un puesto de trabajo o similar.
Elgg	Motor de desarrollo para sitios web sociales.
MaMaS-tng	Motor de razonamiento para Lógica Descriptiva.
Oferta	Una oferta es un anuncio presentado por una persona o institución con el objetivo de ofrecer un puesto de trabajo o similar.
eLaboratory	El eLaboratory de AFRICA BUILD Portal es un espacio donde los usuarios cuentan con las principales herramientas del sitio, que ellos emplean.
Widgets	Los Widgets son elementos ubicados en el eLaboratory para mostrar información de los distintos plugins.

Tabla 1: Definiciones

- Acrónimos:

Acrónimo	Definición
ABP	AFRICA BUILD Portal
API	Application Programming Interface
CSRF	Cross-site Request Forgery
ERS	Especificación de Requisitos Software
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
JSDK	Java Software Development Kit
JVM	Java Virtual Machine
SGBD	Sistema de Gestión de Bases de Datos
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol / Internet Protocol
XML	eXtensible Markup Language

Tabla 2: Acrónimos

- Abreviaturas:

Abreviatura	Definición
Std.	Estándar

Tabla 3: Abreviaturas

4.1.1.4 Referencias

IEEE Recommended Practice for Software Requirements Specification. ANSI/IEEE std. 830, 1998.

4.1.1.5 Visión general del documento

Esta especificación de requisitos que forma parte del capítulo 4, "Análisis del sistema", está dividida en tres secciones:

- Introducción: Sección actual que presenta una visión global de la ERS y del sistema.

- Descripción general: Sección siguiente que describe los factores que afectan al sistema y proporciona una descripción de este, con el fin de conocer las principales funciones que debe realizar, sin entran en detalles.

- Requisitos específicos: Sección final donde se definen con detalle los requisitos que debe satisfacer el sistema.

4.1.2 Descripción general

4.1.2.1 Perspectiva del producto

El sistema a desarrollar forma parte de un sistema mayor, el sistema ABP, y en consecuencia su desarrollo está influenciado y requerido por este. Sin embargo el sistema podría funcionar de manera autónoma aplicándole ligeras modificaciones.

Las interfaces entre el sistema y ABP son las interfaces gráficas y de comunicaciones, las cuales están implementadas por el plugin desarrollado como parte de este sistema para poder ser accedido desde ABP, de hecho, esta es la dependencia que tiene el sistema para no ser independiente, ya que sin esas interfaces no se pueden crear ni ofertas ni demandas por lo que el sistema no realizaría ninguna función.

4.1.2.2 Funciones del producto

El sistema a desarrollar ofrecerá las siguientes funciones:

- Soporte para la gestión de ofertas y demandas por parte de los usuarios de ABP. Esta gestión supone la creación, edición y eliminación de sus propias ofertas y demandas.

- Realización de búsquedas de ofertas y demandas a partir de parámetros de búsqueda.

- Realización de recomendaciones automáticas a las ofertas y demandas de los usuarios basándose en las ofertas y demandas existentes en el sistema.

- Integración del plugin para el ABP con otros componentes del ABP como notificaciones, actividad o sistema de Widgets.

4.1.2.3 Características de los usuarios

El sistema no requerirá de ningún tipo de mantenimiento especial para su correcto funcionamiento más allá del propio mantenimiento de ABP, por lo que los usuarios contemplados para este sistema son derivados de ABP, identificándose los siguientes tipos de usuarios:

- Administrador: Este usuario administrador se refiere al mismo administrador de ABP, que no es más que un usuario que tiene el rol de administrador en la red social. Este usuario tiene a su disposición todas las funcionalidades que tiene un usuario normal pero además puede configurar diversos parámetros de la red social como añadir nuevas funcionalidades añadiendo plugins, o configurando características específicas de estos plugins. Individualizando estos privilegios del administrador al plugin correspondiente a este sistema, el administrador podrá configurar parámetros relativos a la configuración de la base de datos o de las ontologías, además podrá administrar todas las ofertas y demandas existentes en la red social, esto se refiere a que puede editarlas o eliminarlas.

- Usuarios comunes: Estos usuarios son todos aquellos que usaran la red social, como es lógico este tipo de usuario engloba un gran espectro de posibles perfiles de usuario, sin embargo, principalmente ABP está dirigida a estudiantes e investigadores en el dominio de la salud. Pero, por otro lado, debido al potencial de la plataforma, esta puede llegar a incrementar ese espectro y contar con nuevos usuarios con perfiles totalmente distintos. No entrando a día de hoy en estos posibles futuros usuarios y considerando los actuales se pueden definir como usuarios con un perfil académico en el ámbito de la salud y sin conocimiento técnicos de informática, sin embargo se diferencia entre aquellos

que pueden tener un nivel básico en manejo de ordenadores y en aquellos que no tengan ningún tipo de conocimiento en manejo de ordenadores.

4.1.2.4 Restricciones

En este apartado se presentan las restricciones a las que el sistema final se encuentra sujeto. Las restricciones identificadas son dos relativas a las versiones de elementos software sobre los que operan.

Por un lado, pese a que una parte del sistema está desarrollada en el lenguaje Java, lo que supone que el sistema es completamente independiente de la plataforma y del sistema operativo sobre el que se ejecuta, se deberá contar con el intérprete adecuado para cada plataforma y sistema operativo para poder ejecutar los "bytecodes" generados por el compilador Java, es decir contar con la máquina virtual de Java adecuada. Y además que la versión de su JSDK sea la 1.7 o superior.

Por otro lado, dado que otra parte fundamental del sistema es el plugin desarrollado para ABP y que esta está desarrollada mediante Elgg, el funcionamiento del plugin está restringido a la versión 1.8 de Elgg o al soporte que se ofrezca de las funciones actuales en futuras versiones de Elgg.

4.1.2.5 Suposiciones y dependencias

Se supone que una vez descritos los requisitos definitivos del sistema, estos permanecerán estables, y cualquier cambio en los mismos se hará mediante un procedimiento controlado, reflejándose en los documentos oportunos.

Se supone que las posibles futuras actualizaciones que se realicen en las ontologías no cambiaran su formato, es decir, seguirán expresándose en formato DIG y además su estructura de representación permanecerá invariable.

Se supone que los usuarios crearan ofertas y demandas expresando su información en inglés.

La dependencia más importante que presenta el sistema es con MaMaS-tng. Esta dependencia está en que si MaMaS-tng deja de dar servicio, cambia su sintaxis, o sus operaciones, el módulo de recomendación del sistema quedaría totalmente inoperativo.

La otra dependencia que presente el sistema es con la API "Google API Spelling Java". Aunque se trate de una API compilada con todo el código del sistema lo que cualquier cambia futuro no le afectaría, esta API presenta la peculiaridad de que emplea el servicio Google Spell Checker, lo que supone que cualquier cambio en este servicio puede afectar al funcionamiento del sistema. Sin embargo, el incorrecto funcionamiento de esa parte no afectaría en gran medida al sistema e incluso podría ser imperceptible.

4.1.3 Requisitos específicos

En este apartado se detallan los requisitos funcionales que el sistema deberá satisfacer. Todos los requisitos presentados en esta sección son esenciales, es decir, la aplicación no será aceptable en el caso de no cumplir alguno de los requisitos expuestos. Los requisitos se han especificado en base al criterio de testeabilididad, dado un requisito deberá ser fácilmente demostrable si el sistema lo satisface o no.

4.1.3.1 Interfaces externas

En este apartado se detallan los requisitos que afectan a la interfaz de usuario, interfaz con otros sistemas (hardware y software) e interfaces de comunicaciones.

4.1.3.1.1 Interfaces de usuario

- REQ#01: La interfaz de usuario deberá ser accesible a través de un navegador web. Este requisito es matizado por el REQ#06.

- REQ#02: Todos los datos e información deberán mostrarse utilizando páginas y formularios HTML.

- REQ#03: La interfaz de usuario deberá ser consistente con el estilo de ABP derivado de Elgg, es decir, presentará los mismos tipos de letras, colores, estilo de botones o campos, etc.

- REQ#04: Todo el texto presentado en las páginas y formularios deberá estar escrito en algún idioma de entre los que APB ofrezca soporte y que el usuario podrá configurar. Excepto, por un lado, los datos que los usuarios introduzcan para definir las ofertas y demandas que como se indicó en una suposición, solo será admitido en inglés y por otro lado los datos que son generados a partir del contenido de las ontologías, las cuales están en inglés.

4.1.3.1.2 Interfaces Software

- REQ#05: El sistema deberá emplear mensajes DIG 1.1 para el intercambio de datos con MaMaS-tng.

- REQ#06: El sistema deberá funcionar en diferentes plataformas software como Windows, Mac Os, Linux, etc.

- REQ#07: El sistema deberá funcionar de manera óptima sobre los navegadores Google Chrome y Mozilla Firefox y de manera más reducida en cuanto a sus propiedades gráficas en navegadores como Internet Explorer.

4.1.3.1.3 Interfaces de Hardware

- REQ#08: El sistema funcionará sobre diferentes plataformas hardware como Intel, AMD o Sparc y con diferentes arquitecturas hardware (32 y 64 bits).

4.1.3.1.4 Interfaces de comunicación

- REQ#09: El sistema empleará el protocolo de comunicación por Internet TCP/IP, HTTP.

- REQ#10: Para la representación e intercambio de información mediante la comunicación, indicada en el requisito anterior, se utilizará el estándar XML.

4.1.3.2 Funciones

En este apartado se detallan las funciones que debe realizar el sistema para su correcto funcionamiento. Estos requisitos se han listado según los objetivos de los requisitos, entendiendo, un objetivo como un servicio que el sistema ofrece.

4.1.3.2.1 Gestión de demandas

- REQ#11: Los usuarios podrán registrar demandas rellenando un formulario web con una serie de campos que son detallados en el "Modelo de dominio" del Capítulo 5. Este formulario incluirá importantes funciones de ayuda como autocompletado de campos, campos seleccionables y validación de campos.

- REQ#12: Los usuarios podrán editar sus demandas a través del mismo formulario, empleado para su creación. Este formulario deberá cargar y mostrar en cada campo la información actual de la demanda, pudiendo editar cualquier campo excepto el del propietario de la demanda.

- REQ#13: Los usuarios podrán eliminar sus demandas. Esta eliminación supone la pérdida total de la información asociada a su demanda y no será posible su recuperación en un futuro.

- REQ#14: Los usuarios podrán ver todas las demandas pertenecientes a ellos mismos y a sus amigos en la red social. Mostrando estas en forma de dos listas independientes, una lista con las suyas propias y otra lista con las de sus amigos. Esta lista mostrará una versión resumida de las demandas.

- REQ#15: Los usuarios podrán añadir comentarios a cualquier demanda y marcarla como que le gusta.

- REQ#16: Las demandas tendrán una vista principal, que mostrará toda la información de ésta definida por el usuario, así como todos los comentarios que haya recibido.

- REQ#17: Las demandas podrán ser editadas y eliminadas por el usuario administrador, en cualquier momento.

4.1.3.2.2 Gestión de ofertas

- REQ#18: Los usuarios podrán registrar ofertas rellenando un formulario web con una serie de campos que son detallados en el "Modelo de dominio" del Capítulo 5. Este formulario incluirá importantes funciones de ayuda como autocompletado de campos, campos seleccionables y validación de campos.

- REQ#19: Los usuarios podrán editar sus ofertas a través del mismo formulario, empleado para su creación. Este formulario deberá cargar y mostrar en cada campo la información actual de la oferta, pudiendo editar cualquier campo excepto el del propietario de la oferta.

- REQ#20: Los usuarios podrán eliminar sus ofertas. Esta eliminación supone la pérdida total de la información asociada a su oferta y no será posible su recuperación en un futuro.

- REQ#21: Los usuarios podrán ver todas las ofertas pertenecientes a ellos mismos y a sus amigos en la red social. Mostrando estas en forma de dos listas independientes, una lista con las suyas propias y otra lista con las de sus amigos. Esta lista mostrará una versión resumida de las ofertas.

- REQ#22: Los usuarios podrán añadir comentarios a cualquier oferta y marcarla como que le gusta.

- REQ#23: Las ofertas tendrán una vista principal, que mostrará toda la información de ésta definida por el usuario, así como todos los comentarios que haya recibido.

- REQ#24: Las ofertas podrán ser editadas y eliminadas por el usuario administrador, en cualquier momento.

4.1.3.2.3 Realización de búsquedas

- REQ#25: Los usuarios podrán realizar búsquedas de ofertas y demandas. Estas búsquedas recuperaran todas las ofertas y/o demandas que existan en el sistema

independientemente de a quien pertenezcan. Las ofertas y/o demandas se podrán filtrar por país y etiquetas.

4.1.3.2.4 Realización de recomendaciones

- REQ#26: El sistema generará recomendaciones automáticas para las ofertas de los usuarios acorde a las demandas existentes en el sistema.

- REQ#27: El sistema generará recomendaciones automáticas para las demandas de los usuarios acorde a las ofertas existentes en el sistema.

- REQ#28: Las recomendaciones serán producidas mediante un razonamiento semántico basado en cruzar distintos parámetros de la oferta y demanda.

- REQ#29: El sistema formalizará las ofertas y demandas, escritas en Lenguaje Natural, a Lógica Descriptiva empleando técnicas de procesamiento de Lenguaje Natural.

- REQ#30: El sistema realizará una corrección ortográfica de las ofertas y demandas en el proceso de formalización.

- REQ#31: El sistema deberá soportar la sinonimia para formalizar de la misma manera conceptos sinónimos o equivalentes.

- REQ#32: El sistema deberá formalizar parte de información de las ofertas y demandas analizando semánticamente lo que se quiere expresar.

- REQ#33: Cada recomendación expresará el grado de afinidad o proximidad entre la oferta y demanda mediante un porcentaje entre 0% y 100%.

- REQ#34: Cada recomendación ofrecerá una explicación en lenguaje natural a la afinidad o proximidad entre la oferta y la demanda. Esta explicación indicará que requisitos o preferencias no son dados y porqué, por la contraparte.

- REQ#35: Los usuarios podrán ver todas las recomendaciones realizadas sobre sus ofertas y demandas. Teniendo por cada oferta o demanda una lista ordenada de mayor a menor afinidad con las demandas u ofertas recomendadas, acompañadas por el grado de afinidad y un breve resumen de la recomendación.

- REQ#36: Los usuarios podrán ver detalladamente cada recomendación, mostrándole el grado de afinidad, la explicación en lenguaje natural y la oferta y demanda asociadas a la recomendación, resaltando sus campos en conflicto.

4.1.3.2.5 Integración

- REQ#37: El sistema deberá integrase con la actividad del ABP, de manera que cuando se registre una oferta o demanda, se añadirá una entrada en esta actividad con el aviso de la creación de esta nueva oferta o demanda.

- REQ#38: El sistema deberá integrase con el módulo de notificaciones a los usuarios de ABP, de manera que cuando se registre una oferta o demanda, se registre y envié la notificación de esta creación a ciertos usuarios.

- REQ#39: El sistema dispondrá de un Widget, para el eLaboratory de ABP, que mostrará las últimas ofertas creadas en el sistema, permitiendo filtrarlas por país, etiquetas y número a mostrar.

- REQ#40: El sistema dispondrá de un Widget, para el eLaboratory de ABP, que mostrará las últimas demandas creadas en el sistema, permitiendo filtrarlas por país, etiquetas y número a mostrar.

- REQ#41: El sistema dispondrá de un Widget, para el eLaboratory de ABP, que mostrará las últimas recomendaciones realizadas sobre ofertas y/o demandas del usuario, permitiendo filtrarlas por tipo de recomendación y numero a mostrar.

4.1.3.3 Requisitos de rendimiento

- REQ#42: El almacenamiento de la información deberá ser distribuido para evitar la carga de datos y de operaciones en la base de datos interna de ABP.

4.1.3.4 Restricciones de diseño

- REQ#43: El sistema deberá se diseñado de manera que las máquinas de los usuarios realicen el menor trabajo posible en lo relativo a la formalización y recomendación, siendo la maquina servidora quien se ocupe de eso.

4.1.3.5 Atributos del sistema

En esta sección se detallan las propiedades que miden la calidad del sistema:

- Seguridad: El sistema asegura la total protección de la información que maneja ya que esta se encuentran en la base de datos. Esta base de datos únicamente

puede ser accedida por el usuario administrador o usuarios autorizados, lo cual es controlado por el SGBD.

Por otro lado el sistema está protegido de posibles ataques de inyección de código SQL o ataques CSRF que se podrían realizar mediante los formularios HTML. Esta protección es ofrecida mediante funcionalidades del core de Elgg.

- Fiabilidad: El sistema será altamente fiable siempre y cuando el usuario introduzca correctamente los datos, y los servicios externos empleados estén disponibles, según como se indicó en el apartado de "Suposiciones y dependencias".

- Mantenibilidad: El desarrollo del sistema debe basarse en estándares y convenciones ampliamente establecidas, de forma que el desarrollo futuro del mismo no se vea condicionado o dificultado por el uso de técnicas y/o tecnologías propietarias y pueda ser continuado partiendo de la solución aquí propuesta.

  Además, no se han especificado tareas de mantenimiento específicas a este sistema si no que corresponden a las tareas de mantenimiento del ABP. Pero sí existe la posibilidad de aumentar su funcionalidad mediante la actualización de las ontologías o la mejora de algún módulo de los que se compone.

- Portabilidad: El sistema desarrollado será completamente independiente tanto de la plataforma de ejecución como del sistema operativo en el que se ejecute. Esta cualidad es proporcionada por el uso del lenguaje Java gracias a su máquina virtual JVM.

4.1.3.6 Otros requisitos

- REQ#44: El usuario administrador podrá configurar diversos parámetros de configuración del sistema dese la Interfaz de administración de ABP.

- REQ#45: El usuario administrador podrá actualizar las ontologías, que dan soporte al proceso de razonamiento desde la Interfaz de administración de ABP.

- REQ#46: El sistema generará un fichero de log en el cual registrará posibles errores que se puedan producir e información de control, indicando fecha, módulo que lo produce y descripción.

Capítulo 5

DISEÑO E IMPLEMENTACIÓN

En este capítulo se presenta detalladamente el diseño realizado del sistema y su posterior implementación de acuerdo a la solución propuesta acorde al análisis del sistema realizado presentado en el capítulo anterior. En cada apartado se explicarán las decisiones de diseño tomadas y sus detalles de implementación.

Primeramente se presenta una visión global del sistema, que muestra su arquitectura y funcionamiento, con el objetivo de explicar cómo funciona a grandes rasgos el sistema y cómo actúan conjuntamente los distintos módulos que componen el sistema y que serán detallados posteriormente.

5.1 Arquitectura global del sistema

Como se indicó en el apartado "Solución propuesta" del Capítulo 1, el sistema se puede descomponer en cuanto grandes módulos. Estos módulos se pueden distinguir claramente debido a su magnitud y su cometido. Estos cuatro módulos son: Las ontologías, el Preprocesador del Match, el Procesador del Match y la Interfaz gráfica. A continuación se muestra el esquema de la arquitectura global del sistema, en el cual se pueden ver cada uno de estos módulos. Además de esos módulos, también se presenta en este esquema y en la posterior descripción de cada uno de ellos, la base de datos que da soporte al sistema para almacenar la información.

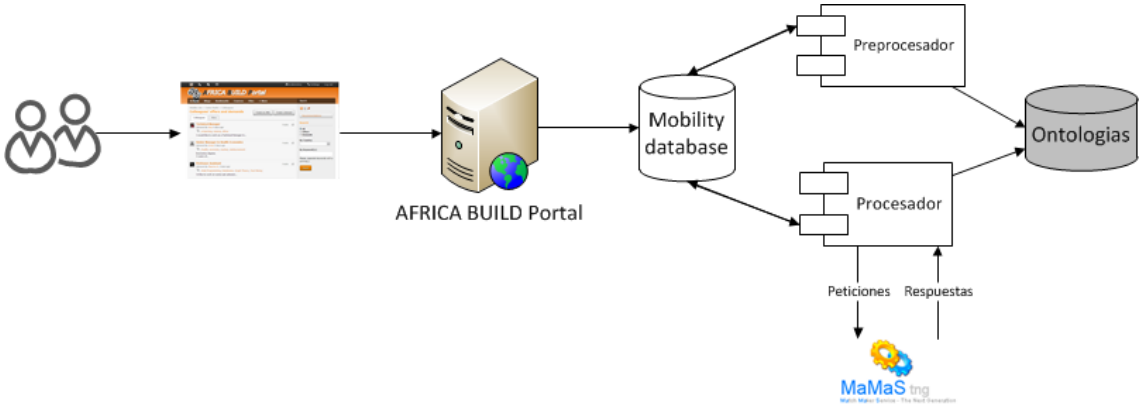


Figura 5-1: Esquema de la arquitectura global del sistema con todos sus módulos

Antes de entrar en detalles de implementación, para comprender este esquema general, el funcionamiento del sistema se puede resumir en:

1. Un usuario crea una oferta o demanda a través de la Interfaz de usuario de AFRICA BUILD Portal.

2. La información de esa oferta o demanda es almacenada en la base de datos distribuida y al mismo tiempo se lanza el módulo preprocesador. Este módulo realiza dos tareas, por un lado registra a esta oferta o demanda para ser tomada en cuenta en el proceso de recomendación y formaliza la oferta o la demanda a una representación lógica.

3. Asíncronamente con el anterior modulo, el modulo procesador entra en ejecución cada un cierto intervalo de tiempo fijado. Este módulo realiza todas las recomendaciones nuevas que haya posible en el sistema a partir de las ofertas y demandas que hayan sido registradas por el modulo preprocesador. La realización de las recomendaciones se basa en realizar inferencias lógicas sobre los pares a recomendar para determinar su proximidad y una justificación a esta proximidad. Las inferencias lógicas son provistas por el razonador MaMaS-tng, que es consultado por el modulo.

4. Finalmente, cada usuario puede ver sus recomendaciones a través de la Interfaz de usuario de AFRICA BUILD Portal. Esta interfaz muestra listadas sus recomendaciones a sus ofertas y demandas ofreciéndole una clasificación por cada oferta o demanda de las mejores demandas u ofertas.

5.2 Módulos del sistema

5.2.1 Base de datos

5.2.1.1 Modelo de dominio

En esta sección se presenta el modelo de dominio de la información con la que trata el sistema. Para dar soporte a la información de este modelo es necesaria una base de datos, la cual es detallada a continuación de esta sección.

El sistema se compone de tres objetos o entidades de información bien diferenciadas, que son: Oferta, Demanda y Recomendación. La formalización de estas entidades no se ha realizado desde cero, en este Trabajo, sino que se ha partido del modelo de dominio tratado en INFOBIOMED [5]. La formalización presentada a continuación es un refinamiento de ese modelo con respecto al entorno específico de este sistema.

- Oferta: La oferta es el objeto que representa a una solicitud en la que se requiere de personal para cubrir un puesto de trabajo y es realizada por una persona física o jurídica. Una oferta se define a partir de los siguientes elementos de información:

Campo	Descripción
Titulo	El título define brevemente acerca de que trata la oferta, por ejemplo puede indicar el puesto de trabajo.
Descripción	En la descripción se presenta detalladamente la oferta aportando información acerca de la institución ofertante y describiendo las tareas a llevar a cabo en el puesto ofertado.
Institución	Nombre de la institución que realiza la oferta.
País	País donde se oferta el puesto.
Categoría	La categoría indica el puesto de trabajo ofertado.
Campo de aplicación	Especifica la categoría en el caso de algunas categorías que pueden tener diversas aplicaciones.
Palabras clave	Lista de palabras clave que representan a la oferta.
Duración	Duración que toma el puesto ofertado.
Fecha de comienzo	Fecha que indica el comienzo del puesto ofertado.
Educación y experiencia	Conocimientos educativos y experiencia que son requeridos a los candidatos que quieran optar al puesto.
Competencias del candidato	Capacidades y competencias, excluyendo las anteriores, que son requeridas a los candidatos que quieran optar al puesto.
Requisitos de la aplicación	Cualquier tipo de requisito que no se pueda contemplar en los tipos anteriores.
Alojamiento	Información que indica si además se ofrece alojamiento junto con la oferta.
Salario	Remuneración económica que es ofrecida por el puesto.
Comentarios	Cualquier comentario que se desee realizar para especificar algo más la oferta y que no esté relacionado con ningún campo anterior.

Fecha de validez	Fecha tras la cual la oferta dejará de tener validez.
Persona de contacto	Persona con la que contactar en caso de estar interesado en la oferta.
Email de contacto	Email de contacto de la persona anterior.
Página web de la institución	Página web de la institución que realiza la oferta.

Tabla 4: Descripción del modelo de dominio de la oferta

- Demanda: La demanda es el objeto que representa a una solicitud en la que una persona presenta sus conocimientos, competencias y preferencias de cara a solicitar un puesto de trabajo. Una demanda se define a partir de los siguientes elementos:

Campo	Descripción
Titulo	El titulo define brevemente acerca de que trata la demanda, por ejemplo se puede indicar la profesión natural o rol del demandante y que busca.
Descripción	En la descripción se presenta detalladamente la demanda aportando información acerca de que busca el demandante.
País	País o países donde el demandante busca empleo.
Categoría	La categoría indica el puesto de trabajo que busca el demandante.
Campo de aplicación	Especifica la categoría en el caso de algunas categorías que pueden tener diversas aplicaciones.
Palabras clave	Lista de palabras clave que representan a la oferta.
Características deseadas	Características que el demandante busca que tenga la institución y el trabajo buscado.
Características propia	Características propias del demandante que lo definen.
Duración	Duración deseada para el puesto que busca.
Fecha de comienzo	Fecha deseada para el comienzo del puesto que busca.
Educación y experiencia	Conocimientos educativos y experiencia que tiene el demandante.
Intereses	Descripción resumida de los principales intereses que tiene el demandante.
Salario	Remuneración económica que busca el demandante.
Alojamiento	Información que indica si el demandante busca un puesto que incluya el alojamiento.

Tabla 5: Descripción del modelo de dominio de la demanda

- Recomendación: La recomendación es el objeto que presenta, para una oferta o demanda, una demanda u oferta que es similar en términos de que cumple con los requisitos solicitados o pedidos. Una recomendación se define a partir de los siguientes elementos:

Campo	Descripción
Solicitud	La solicitud es el elemento para el cual se busca recomendación, esta solicitud puede ser tanto una oferta como una demanda.
Propuesta	La propuesta es el elemento recomendado a la solicitud, esta propuesta puede ser tanto una oferta como una demanda en función a la solicitud

Proximidad	Medida cuantitativa que indica la proximidad entre la solicitud y la propuesta.
Explicación	Explicación en lenguaje natural del grado de la medida de proximidad entre la solicitud y la propuesta.

Tabla 6: Descripción del modelo de dominio de la recomendación

5.2.1.2 Modelo entidad relación

El modelo de dominio anterior es soportado por una base de datos que almacena toda la información. Esta base de datos se ha diseñado partiendo del siguiente modelo Entidad-Relación que representa al modelo de dominio.

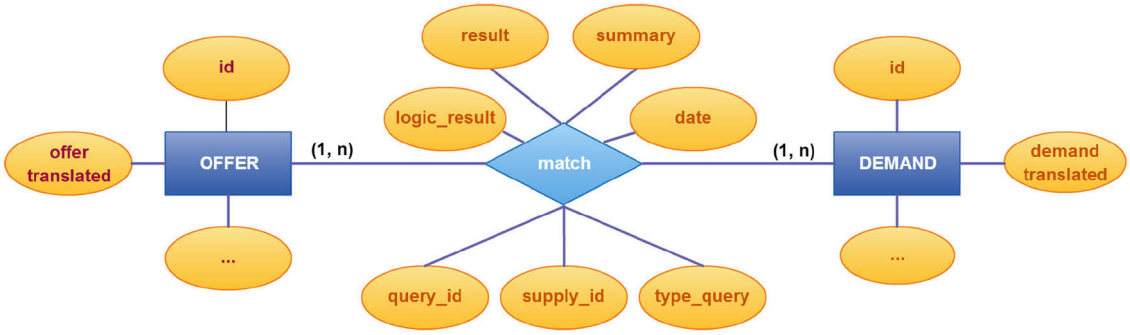


Figura 5-2: Modelo entidad relación de la base de datos del sistema

El modelo cuenta con dos entidades, OFFER y DEMAND que representan a los objetos Oferta y Demanda, respectivamente. En el anterior modelo, no se han representado todos los atributos que tienen, estos atributos no mostrados son los mismos campos o elementos que se definieron en el modelo de dominio. Los atributos que se muestran aquí son el "id" que es un identificador numérico tanto para la oferta como demanda y los campos "offer_translated" y "demand_translated" que son campos para almacenar la formalización lógica, realizada por el módulo preprocesador. Como se muestra en el diagrama, además de esas dos entidades se tiene la relación "match" la cual representa a las recomendaciones. Los atributos de esta relación son: "query_id" e "supply_id que referencian los elementos que actúan como solicitud y propuesta, "type_query" que identifica si la "query_id" pertenece a una oferta o demanda, "ranking" que expresa la medida de proximidad, "logic_result" que contiene el resultado lógico de realizar la recomendación, "summary" que contiene una explicación muy breve en lenguaje natural de la recomendación, "result" que contiene la explicación de la recomendación y "date" que indica la fecha de realización de la recomendación.

La implementación final de este modelo ha derivado en una base de datos distribuida pero que viéndola de forma global sigue este modelo.

5.2.1.3 Implementación distribuida

La implementación distribuida del anterior modelo se basa en que algunos atributos de las entidades OFFER y DEMAND se encuentran en la propia base de datos de ABP y otros atributos en una base de datos externa.

Por un lado para almacenar los atributos en la base de datos de ABP se hace uso de las entidades de Elgg que abstraen el manejo del modelo de datos interno, en concreto cada vez que se inserta una oferta o demanda se crea una entidad del tipo ElggObject especificando su subtipo como "offer" o "demand" y contiene como metadatos los atributos "Titulo", "Descripción", "Palabras clave" y un identificador que almacena el identificador numérico de esa misma oferta o demanda en la base de datos externa, lo que permite relacionar las dos partes de la misma oferta o demanda.

Por otro lado la base de datos externa es una base de datos MySQL que contiene las tres tablas correspondientes al modelo entidad relación anterior, es decir, las tablas "offers", "demands" y "matches". Donde, obviamente las tablas "offers" y "demands" tienen todos los atributos presentados en el modelo entidad relación a excepción de los citados anteriormente para la base de datos interna de ABP.

La justificación de implementar esta solución hibrida o distribuida es la siguiente: Por un lado, como se define en el requisito REQ#42, se buscaba reducir la carga de acceso sobre la base de datos interna de ABP, realizándose así solamente operaciones de lectura o escritura por parte del plugin del Mobility pero no por parte del módulo de preprocesado y procesado, lo que deriva en la necesidad de contar con la base datos externa; por otro lado se busca tener un tiempo de respuesta bajo para la visualización rápida de las ofertas y demandas, lo que deriva en tener la información más básica de ellas en la base de datos interna y no tener que realizar peticiones a la externa.

5.2.2 Ontologías

En esta sección se presenta el desarrollo e implementación de las Ontologías que forman parten del sistema. Estas ontologías son un elemento clave en el sistema dado el sentido semántico de este, por lo que se consideran como un módulo más del sistema.

Como se ha comentado anteriormente e incluso indica el título del Trabajo, estamos ante un sistema semántico, el cual basa su funcionamiento en una semántica o razonamiento definido.

Este funcionamiento semántico es establecido por las Ontologías, las cuales definen un vocabulario formal que modela y describe el dominio del sistema y que establece la semántica acerca de este dominio para poder razonar sobre él. Entendiendo por el dominio del sistema a la gestión de recursos humanos en el entorno de la Salud y África. En resumen, estas ontologías son empleadas para crear las formalizaciones lógicas de las ofertas y demandas y para generar las recomendaciones siguiendo la semántica que establecen.

5.2.2.1 Proceso de desarrollo de las Ontologías

Dada la magnitud en cuanto al tamaño e importancia de estas ontologías, para su desarrollo se ha seguido la metodología para el desarrollo de ontologías "Methontology". La decisión de seguir esta metodología se fundamente en que Methontology es una de las metodologías más detallada y especificada algo conveniente dada la poca experiencia en el desarrollo de ontologías. Por otro lado Methontology define extensamente su etapa de conceptualización, la cual se ha seguido para el desarrollo de las ontologías. A continuación se presenta el proceso de desarrollo:

1. Adquisición de conocimiento. Como primer paso antes de comenzar con las etapas de conceptualización, está la adquisición de conocimiento. Esta adquisición supone recopilar fuentes de información, referentes al dominio, para ser usadas por las etapas de conceptualización. Concretamente, esta etapa ha supuesto recuperar ofertas y demandas relacionadas con la Salud y África, para ello se han seguido dos procedimientos. Por un lado se pidió la colaboración a miembros de otras instituciones del proyecto AFRICA BUILD, a los cuales se les pidió que crearan una serie de ofertas y demandas inventadas pero ajustadas a situaciones reales. Y por otro lado se realizó una búsqueda de ofertas y demandas en portales de empleo orientados a la salud o de África, los portales consultados han sido: www.medicaljobs.org, www.imrmedical.com, www.jobs4medical.co, www.bioinformatics.org y www.findajobinafrica.com. De esta recolección de ofertas y demandas se obtuvieron 112 ofertas y demandas, las cuales han sido usadas por las siguientes etapas de

conceptualización y para el diseño del analizador de lenguaje natural del módulo preprocesador.

2. Construcción del glosario de términos. Este paso es el primero de la fase de conceptualización de las ontologías. El objetivo en este paso es elaborar un glosario de términos que incluya términos relevantes del dominio extraídos de las ofertas y demandas recopiladas en el paso anterior. En este glosario de términos por cada término identificado se recoge la siguiente información: Una descripción o definición en lenguaje natural del término, términos sinónimos, posibles acrónimos del término y la frecuencia de aparición. Un ejemplo de uno de estos términos y su representación en el glosario es el siguiente:

Nombre	Sinónimos	Acrónimos	Descripción	Frecuencia
Master of science	Master degree	MS, SM, MSc	An academic degree higher than a bachelor's degree but lower than a doctor's degree	46

Tabla 7: Ejemplo de representación de un término en el glosario de términos

El glosario de términos resultante de esta etapa de extracción de términos cuenta con 174 términos o conceptos.

3. Construcción de taxonomías de términos. A partir de los términos extraídos en el paso anterior, el siguiente paso es categorizar y clasificar los términos en taxonomías. Estas taxonomías definen jerarquías de términos, de manera que los agrupan en subdominios más específicos y categorizan los términos en niveles de términos más o menos generales dentro del subdominio. De esta manera se elaboraron 7 taxonomías relacionadas con las siguientes categorías: Educación y conocimientos, Geografía, Institución, Idiomas, Ocupaciones, Requisitos y Rol. La construcción de estas taxonomías en cuanto a su categorización y términos que las componen han sido discutidas con otros expertos del proyecto AFRICA BUILD para contrastar las distintas posibilidades y desarrollar una solución consensuada.

Tras estas tres primeras etapas de conceptualización, el proceso de desarrollo de las ontologías no ha continuado con las siguientes etapas de conceptualización que marca Methontology. Esto se debe a la naturaleza de las ontologías necesarias para este sistema, las cuales no cuentan con elementos como atributos, constantes o reglas. Aunque si cuentan con roles y axiomas que definen los términos pero su conceptualización está marcada por los requisitos de implementación.

5.2.2.2 Implementación final de las Ontologías

La implementación de las ontologías es la última etapa en el proceso de desarrollo de las ontologías. Esta implementación supone producir el resultado final de las ontologías a partir de la conceptualización anterior pero de manera que cumpla con las restricciones, requisitos y funcionamiento del sistema.

Hasta este punto, con la etapa de conceptualización las ontologías simplemente están formadas por taxonomías de los términos que representan el dominio, sin embargo no dotan de la semántica necesaria para ser empleada por el módulo de razonamiento o procesador. La implementación de estas ontologías ha estado marcada por las siguientes restricciones o decisiones de diseño del sistema:

- La primera cuestión a tratar es identificar qué información de las ofertas y demandas serán empleadas para realizar las recomendaciones, es decir, que campos o elementos serán cruzados. Esto influye en la implementación de las ontologías ya que cualquier información a emplear por las recomendaciones debe estar definida de alguna manera en las ontologías.

- La siguiente cuestión es el hecho de ofrecer una recomendación dual, es decir que las recomendaciones sean calculadas desde los dos puntos de vista, como ofertantes y como demandantes.

- El último tema es acerca de las restricciones de representación de las ontologías para poder ser utilizadas por el razonador de Lógica Descriptiva "MaMaS-tng".

A partir de estas cuestiones, la solución para la implementación de las ontologías se basa en:

- La información a cruzar se ha establecido en aquella información que de manera simétrica o equivalente aparece tanto en la oferta como en la demanda, es decir, información que puede ser comparada. Así, la información contrastada es acerca de los siguientes elementos: País, Categoría, Campos de aplicación, Duración, Fecha de comienzo, Educación, Salario y Alojamiento. A partir de esta información se han definido roles para poder formalizar la información de las ofertas y demandas. Así, estos roles definen la estructura básica de las ofertas y demandas que es complementada con la terminología del dominio. Además de estos roles, se han definido otros roles los cuales son empleados para definir y describir la terminología de las taxonomías, dotando así de la semántica deseada.

- Para ofrecer las recomendaciones desde los dos puntos de vista puesto que el sistema de recomendación no es simétrico, las ontologías creadas se han

duplicado, contando así, con una ontología para emplearla al realizar las recomendaciones desde el punto de vista del demandante y otra para emplearla al realizar las recomendaciones desde el punto de vista del ofertante. Las diferencias entre estas dos ontologías únicamente está en las definiciones o descripciones de los conceptos, teniendo así las mismas taxonomías y conceptos pero definidos en función a la semántica específica de cada caso.

- Dado que las ontologías serán empleadas por el razonador de Lógica Descriptiva "MaMaS-tng" estas deberán estar expresadas en el formato entendido por este. Por ello las ontologías han sido modeladas empleando el sub-lenguaje de Lógica Descriptiva ALN y expresadas mediante un subconjunto de la sintaxis de DIG 1.1 [34].

Como resultado las ontologías creadas son las siguientes, de las cuales se presenta una breve descripción, numero de conceptos que define, numero de roles primarios y de roles secundarios que define. Los roles primarios son aquellos usados para representar la estructura de las ofertas y demandas y los secundarios aquellos usados para definir o describir los conceptos de las ontologías:

Ontología	Descripción	Conceptos	Roles primarios	Roles secundarios
Geography	Modela información acerca de países y localizaciones geográficas del mundo. Jerarquiza y define estos para ofrecer tres distintas clasificaciones entre las distintas combinaciones de países.	230	hasLocation	isLocatedIn countryCode
Education	Modela información acerca de títulos educativos, conocimientos educativos y sus niveles educativos. Jerarquiza los títulos y conocimientos educativos en distintas ramas y define los títulos a partir de los conocimientos que otorgan.	259	hasEducation hasKnowledge hasEducationLevel	givesKnowledge

Occupation	Modela información acerca de ocupaciones, especializándose en las ocupaciones de Health y Health Informatics. Jerarquiza las ocupaciones en distintos niveles	242	hasOccupation hasField	
Language	Modela los distintos idiomas existentes en el mundo	114	hasLanguage	
Requirement	Modelas distintos requisitos acerca del salario, duración, fechas y alojamiento.	34	hasAccommodation hasDuration hasSalary hasStartDate hasMonth hasYear	accommodation salary

Tabla 8: Descripción resumida de las ontologías del sistema

5.2.3 Preprocesador

El modulo Preprocesador, como su nombre indica y se ha comentado anteriormente, no realiza el procesamiento de las recomendaciones, pero sí que realiza tareas importantes de cara al posterior procesamiento. Es decir, su tarea se basa en preparar la información para poder realizar el procesamiento de las recomendaciones.

Este módulo se encuentra implementado totalmente en Java y hace uso de las tecnologías o librearías OpenNLP, WordNet y Google Spell Checker.

El modulo es ejecutado automáticamente por la Interfaz gráfica cuando se crea o edita una oferta o demanda, procediendo a registrar la oferta o la demanda para ser incluida en el proceso de recomendación y formalizar su información a Lógica Descriptiva. Antes de pasar a explicar estas dos tareas, hay que indicar que estas son realizadas tanto ante la creación como ante la edición ya que al editar una oferta o demanda, obviamente esta habrá cambiado y sus recomendaciones estarán obsoletas.

5.2.3.1 Preparación de Matches

Esta es la primera tarea que se realiza en la etapa de preprocesado y se refiere a añadir la oferta o demanda, creada o editada, en la tabla "matches". Como se describió en la sección de la base de datos, la tabla "matches" contiene las recomendaciones entre las ofertas y demandas existentes en el sistema. Para calcular estas recomendaciones el modulo procesador calcula la proximidad entre pares oferta-demanda, es decir entre cada entrada que se encuentra en esta tabla, pero no tiene ninguna constancia de las ofertas y demandas que existen en el sistema, por ello este módulo se encarga de generar todos los posibles pares entre las ofertas y demandas existentes.

Esta generación no se hace de forma total si no que cada vez que se crea una oferta o demanda se generan únicamente los pares correspondientes a esta oferta o demanda.

Así, cada vez que se crea una oferta o demanda en el sistema se insertan las siguientes entradas en la tabla "matches":

- Entradas en las que la oferta o demanda insertada actúa como elemento a ser recomendado, es decir, la query de la recomendación será esta oferta y demanda. Se inserta una entrada de este tipo por cada elemento contrario (oferta o demanda) al insertado que existan en el sistema.

- Entradas en las que la oferta o demanda insertada actúa como elemento recomendador, es decir será la contraparte de las entradas del tipo anterior. Se inserta una entrada de este tipo por cada elemento contrario (oferta o demanda) al insertado que existan en el sistema.

5.2.3.2 Formalización

La segunda tarea que se realiza en esta etapa de preprocesado es la formalización de la oferta o demanda a su correspondiente descripción de su información mediante Lógica Descriptiva.

El objetivo de la formalización es obtener de la oferta o demanda una representación lógica expresada con un subconjunto de la sintaxis de DIG 1.1, la cual será usada, posteriormente, por el módulo de procesado para definir las instancias de cada oferta y demanda a cruzar. La formalización resultante expresa la información de la oferta o demanda mediante la conjunción de roles y conceptos que han sido definidos en las ontologías. Esta información expresada, únicamente es la concerniente a los parámetros o atributos que serán cruzados, los cuales han sido especificados en la sección de descripción de las Ontologías del sistema.

El algoritmo para la elaboración de esta formalización es el siguiente: Se tienen dos clases distintas que son Oferta y Demanda las cuales tienen como atributos los campos que son cruzados. Cuando es lanzado este módulo, se crea el objeto del tipo Oferta o Demanda pasándole los valores que se han insertado y que se corresponden con esos atributos. A continuación, se llama al método que es el encargado de crear la formalización. El método comprueba si cada uno de estos atributos tiene un valor no vacío para entonces pasar a su formalización concreta. En este punto existen dos posibilidades para la formalización de ese atributo, debido a su naturaleza, las cuales son:

- La formalización del atributo puede ser directa o semidirecta, es decir, su valor se corresponde literalmente con roles o conceptos definidos por las ontologías, o a lo sumo su valor es equivalente a roles o conceptos. Esta situación se da con atributos como el país, el alojamiento o la duración.

- La formalización del atributo no es directa debido a que el atributo es un atributo de texto libre expresado en lenguaje natural donde se puede encontrar cualquier tipo de información, por lo que es necesario realizar un análisis para extraer la información que se espera encontrar en ese atributo. Esta situación se da únicamente con los atributos donde se expresa el conocimiento requerido o poseído según sea en la oferta o demanda.

Esta segunda posibilidad es la más interesante y compleja ya que se realizan distintas tareas de procesamiento de lenguaje natural, por ello se presenta a continuación.

Como primera etapa de la formalización se realiza una corrección ortográfica del texto introducido por el usuario. El objetivo de esta corrección es facilitar a las siguientes etapas el análisis del texto dado que palabras que fueran relevantes pero que estuvieran

mal escritas no serían identificadas. El resultado de esta etapa es el mismo texto corregido.

En la siguiente etapa se realiza un análisis morfosintáctico del texto ya corregido. El primer paso de este análisis es separar el texto en oraciones. Teniendo esta lista de oraciones, se continúa el análisis de cada oración, el cual por una parte trata de identificar el predicado de la oración mediante su análisis sintáctico y posteriormente identifica todos los sintagmas nominales que aparecen en el predicado mediante su análisis morfológico. El objetivo de este análisis es obtener una representación simplificada de la oración que contenga todos sus verbos y sus sustantivos, ya que en la siguiente etapa se intentará identificar los verbos y sustantivos con roles y conceptos de la ontología. El resultado de esta esta etapa es una lista de las oraciones del texto, donde cada oración se compone por una lista de verbos y por un conjunto de sintagmas nominales los cuales se encuentran descompuestos en todos sus sustantivos.

En la posterior etapa se realiza un breve análisis semántico para identificar parcialmente el significado de la oración. Esta identificación parcial se refiere a tratar de reconocer lo que la oración está afirmando o más bien, en el caso de la educación, en reconocer si se está afirmando que se tienen ciertos conocimientos (demanda) o que se requieren tales conocimientos (oferta). Esta identificación se realiza a partir de la categorización verbal, especificada por WordNet, de los verbos que contiene la oración. Wordnet agrupa los verbos que tienen un significado equivalente en 15 categorías, como por ejemplo la categoría “verb.communication” que agrupa algunos verbos como decir, preguntar, ordenar, o la categoría “verb.stative” que agrupa algunos verbos como ser, tener. Así, en el caso de la demanda se identifica si alguno de los verbos pertenece a categorías que expresan posesión y en el caso de la oferta si expresan requerimiento o necesidad.

La siguiente etapa del proceso de formalización es dependiente de la anterior, ya que solo se realiza en el caso de que no exista ningún verbo en la oración o que exista algún verbo que pertenezca a una categoría buscada. El cumplimiento de alguno de estos requisitos indica que quizás se puedan encontrar conceptos que denotan conocimientos educativos. Así, el objetivo de esta etapa es encontrar conceptos o sinónimos de conceptos definidos en la ontología. Primero se busca el literal completo de cada sintagma nominal, encontrado en el análisis morfológico, en la ontología, si no existe se procede a buscar por sinónimos. Esta búsqueda de sinónimos se basa en determinar si el término a buscar puede ser un sinónimo de un concepto representado en la ontología, por ello los conceptos definidos en la ontología presentan una anotación con el número del synset de WordNet al que pertenecen, dado que en WordNet los sinónimos son agrupados en un mismo synset. Así, se realiza una consulta que proporcione el identificador de synset del término a buscar y este identificador es buscado entre las anotaciones de los conceptos. Si tampoco se encuentra, se pasa a realizar el mismo

proceso de búsqueda, primeramente como literal y luego por sinónimo, por cada sustantivo que compone el sintagma nominal. En cualquier caso cuando se encuentra el termino ya sea literalmente o por sinónimo, es devuelto y almacenado el concepto equivalente de la ontología.

Como última etapa del proceso de formalización esta la traducción de todos los conceptos anteriormente encontrados a su representación lógica. Es decir, definir si los conceptos encontrados son relativos a conocimientos o a niveles de conocimientos para expresarlos acompañados del rol correspondiente "hasEducation", "hasKnowledge" o "hasEducationLevel". Esta determinación se basa en consultar a la ontología de cual concepto son descendientes los conceptos encontrados.

Tras tener todos los atributos formalizados y agrupados en una única expresión, esta expresión es almacenada en los atributos "offer_translated" o "demand_translated" de las correspondientes tablas en la base de datos.

5.2.3.3 Clases

A continuación se presentan brevemente las clases Java que componen este módulo.

5.2.3.3.1 Clase PreprocessorMatch

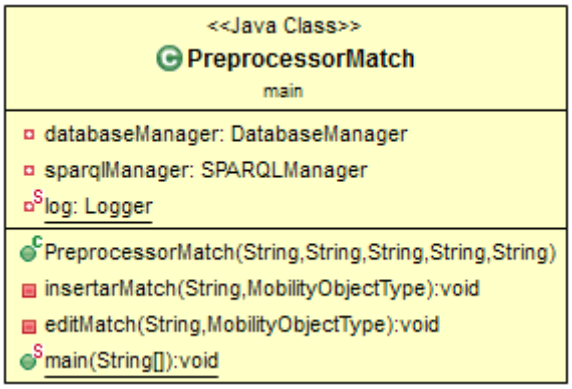


Figura 5-3: Clase PreprocessorMatch

Esta es la clase principal del módulo, la cual realiza el algoritmo de preprocesado. Primeramente comprueba que tipo de objeto tiene que crear (offer o demand), para luego insertar o editar su match correspondiente y posteriormente obtener su formalización llamando al método de formalización del objeto.

5.2.3.3.2 Clase Offer

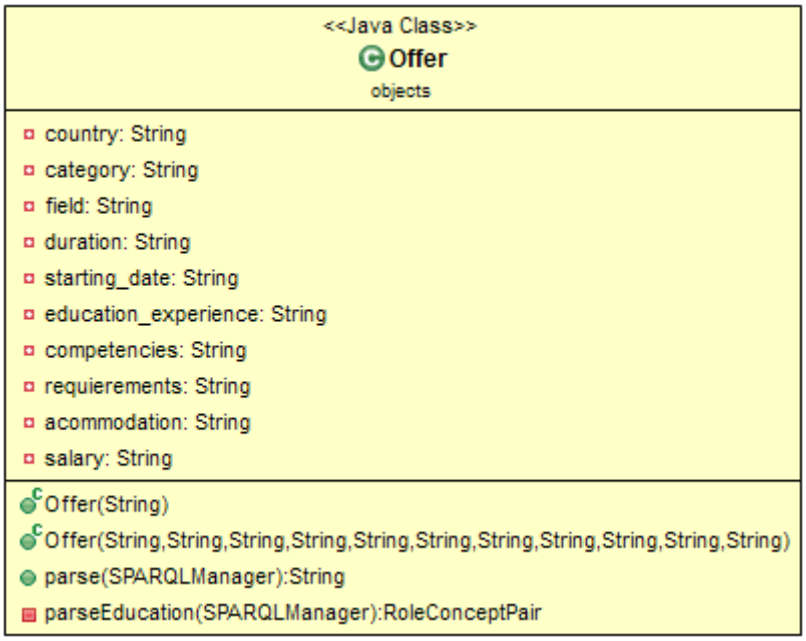


Figura 5-4: Clase Offer

La Clase Offer define los objetos del tipo oferta, los cuales son construidos por la clase principal y es definida con los parámetros a cruzar de una oferta. Tiene como métodos el método "parse" que realiza la formalización de la oferta empleando funciones de la clase DIGParser y el método "parseEducation" que formaliza el campo "education_experience" empleando funciones de la clase ParserUtils.

5.2.3.3.3 Clase Demand

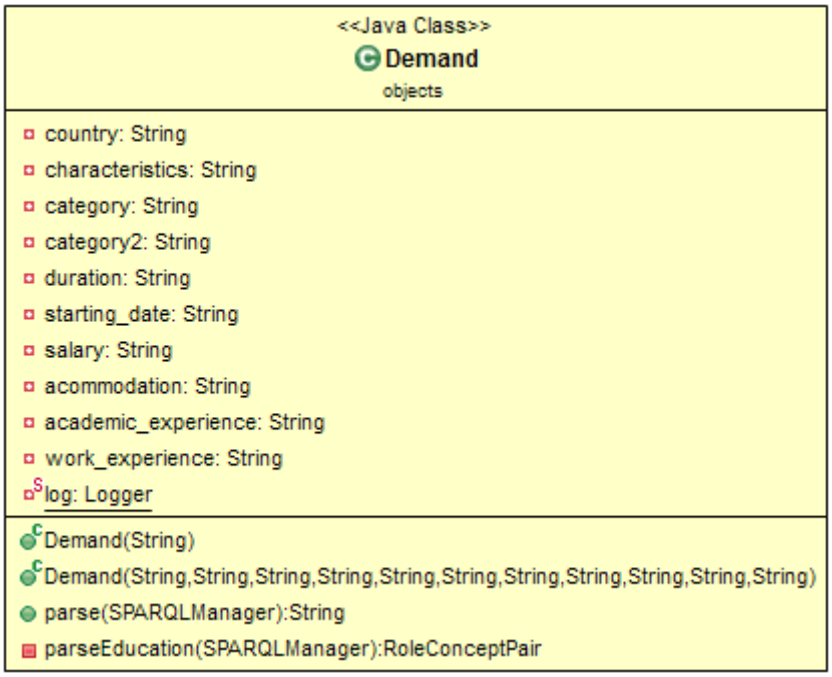


Figura 5-5: Clase Demand

La Clase Demand define los objetos del tipo demanda, los cuales son construidos por la clase principal y es definida con los parámetros a cruzar de una oferta. Tiene como métodos el método "parse" que realiza la formalización de la oferta empleando funciones de la clase DIGParser y el método "parseEducation" que formaliza el campo "academic_experience" empleando funciones de la clase ParserUtils.

5.2.3.3.4 Clase ParserUtils

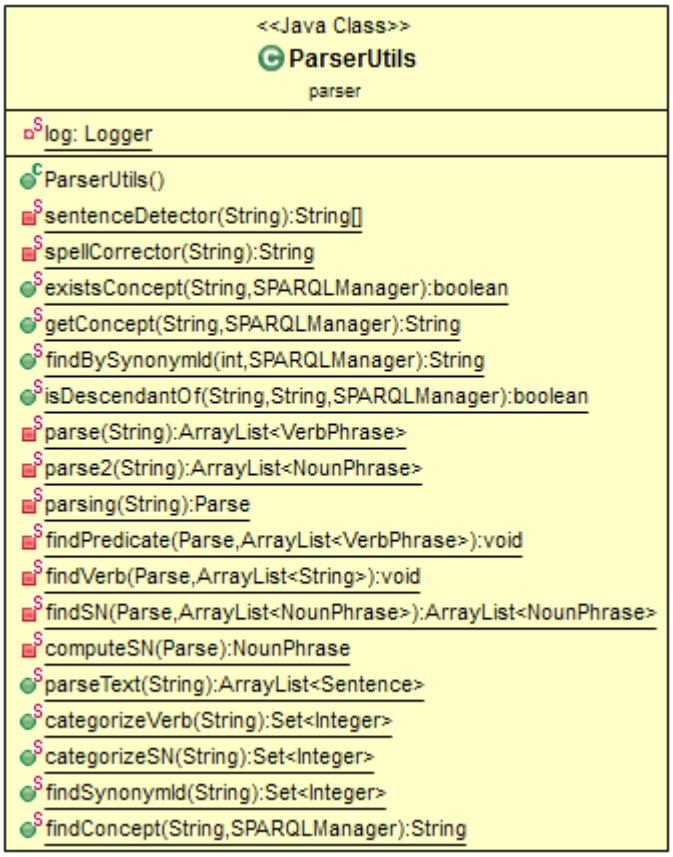


Figura 5-6: Clase ParserUtils

Esta clase contiene una serie de métodos estáticos para el procesamiento de lenguaje natural que son empleados por los objetos de tipo Offer y Demand en la formalización de sus campos de educación. Entre estos métodos están métodos para la corrección ortográfica, el análisis sintáctico, el análisis semántico o la búsqueda en la ontología de un concepto.

5.2.3.3.5 Clase DIGParser

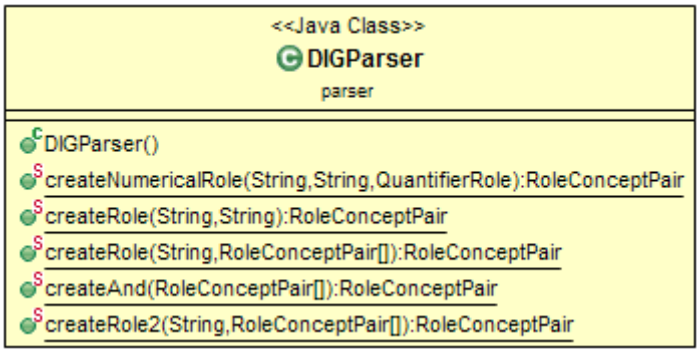


Figura 5-7: Clase DIGParser

Esta clase permite la creación de la representación DIG para la formalización de los roles y conceptos. Emplea la API DIG 1.1 XMLBeans para crear esas representaciones.

5.2.4 Procesador

El modulo procesador podría definirse como el modulo principal o más importante del sistema dado que es el que realiza las recomendaciones, sin embargo, este módulo no funcionaría sin los presentados anteriormente.

El modulo se encuentra desarrollado en Java y es ejecutado automática y periódicamente como un proceso del sistema operativo. Esta configuración de ejecución automática y periódica es definida mediante el manejador que provee Elgg para el cron o administrador de tareas del sistema operativo. De esta manera se puede modificar de manera rápida y sencilla cada cuanto tiempo se ejecutará el módulo.

El objetivo de este módulo es calcular la proximidad y una explicación en lenguaje natural a esa proximidad, de todos los pares de recomendaciones generados por el modulo preprocesador y que todavía no hayan sido calculados. Es decir, calcular las recomendaciones correspondientes a entradas existentes en la tabla "matches" que todavía no se hayan calculado en anteriores ejecuciones de este módulo, bien porque se hayan creado después o porque se hayan editado.

El cálculo de estos atributos de proximidad y explicación de las recomendaciones, se realiza de manera semántica a través de inferencias lógicas provistas por el razonador "MaMaS-tng" y con la base de conocimiento definida por las ontologías desarrolladas que modelan el dominio y la semántica. Por tanto, este módulo simplemente realiza la orquestación de llamadas o servicios al razonador.

5.2.4.1 Algoritmo de recomendación

El algoritmo de recomendación es el encargado de realizar la orquestación de todas las llamadas al razonador "MaMaS-tng" dependiendo de las inferencias que se vayan requiriendo realizar para cada recomendación que se procese.

La comunicación con el razonador es a través de su interfaz DIG a la cual se realizan peticiones HTTP siguiendo el esquema DIG. Toda esta tarea de comunicación y tratamiento de las peticiones y respuestas es modelado por medio de la clase "Reasoner" que abstrae el acceso al razonador "MaMaS-tng".

A continuación, en el siguiente diagrama de flujo, se presenta este algoritmo de recomendación, explicando en cada paso los aspectos de interés de cara al proceso de recomendación omitiendo los detalles de su implementación software.

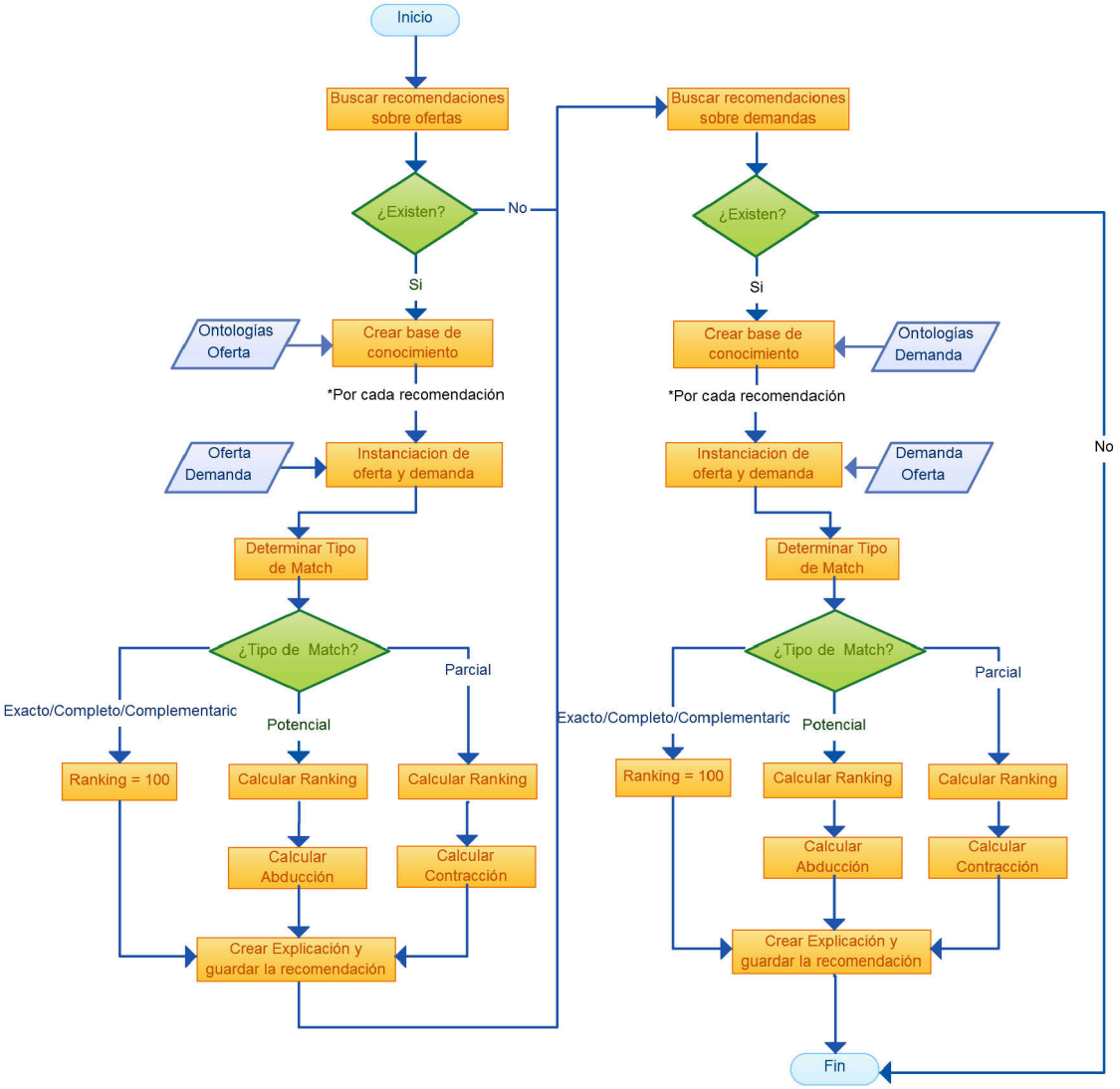


Figura 5-8: Diagrama de flujo del algoritmo de recomendación del sistema

1. El primer paso es buscar recomendaciones a realizar sobre ofertas. Es decir, buscar en la tabla "matches" entradas donde el "type_query" sea "OFFER" y "ranking" este a NULL, lo que significa que esas recomendaciones han sido generadas por el preprocesador pero aún no se han calculado.

2. El siguiente paso, si al menos existe una entrada que cumpla las anteriores condiciones, es generar la base de conocimiento en "MaMaS-tng". Este proceso consiste en solicitar a "MaMaS-tng" la creación de una nueva base de conocimiento en su servidor y definirla con el contenido de las Ontologías que definen la semántica para la recomendación desde el punto de vista de la oferta.

A continuación, por cada entrada que exista, se repite el mismo procedimiento explicado en los siguientes pasos.

3. El siguiente paso es realizar la instanciación de los dos elementos que intervienen en cada recomendación, es decir, la oferta y demanda. Esto supone crear las instancias de esos elementos en la base de conocimiento asignada por "MaMaS-tng" en el anterior paso. Estas instancias se definen a partir de un nombre, para poder identificarlas en futuras consultas y con la formalización lógica de la instancia. El nombrado de la instancias sigue el patrón "Tipo de objeto (OFFER o DEMAND) + ID en la base de datos" y como formalización la formalización creada por el preprocesador para ese elemento y que se encuentra en la base de datos en los campos "offer_translated" y "demand_translated".

4. El siguiente paso es determinar el tipo de match entre la oferta y demanda. Esto se realiza empleando la inferencia "matchType" de "MaMaS-tng", a la cual se le indica las dos instancias a machear, las cuales han sido definidas en el paso anterior. Como resultado, esta inferencia devuelve el tipo de match, el cual puede ser: Exacto, Completo, Complementario, Potencial o Parcial. Dependiendo del tipo devuelto se tienen las siguientes alternativas:

    4.1. Si el tipo devuelto es Exacto, Completo o Complementario significa que en este caso la demanda satisface completamente los requisitos, por lo que su afinidad o ranking se cuantifica en un 100%.

    4.2. Si el tipo devuelto es Potencial significa que la demanda no satisface completamente la oferta. Para conocer cuál es la afinidad o ranking entre oferta y demanda se emplea la inferencia "rankPotential" de "MaMaS-tng" y a continuación para conocer la explicación a esa afinidad se emplea la inferencia "abduce".

    4.3. Si el tipo devuelto es Parcial significa que existen conflictos entre la oferta y la demanda. Para conocer cuál es la afinidad o ranking entre oferta y demanda se emplea la inferencia "rankPartial" de "MaMaS-tng" y a continuación para conocer la explicación a esa afinidad se emplea la inferencia "contract".

5. El siguiente paso, independientemente de cual fuera el tipo devuelto, es generar una explicación en lenguaje natural para la recomendación y actualizar la correspondiente entrada de la recomendación en la tabla "matches" con el ranking y esta explicación. Esta explicación se basa en el tipo de match, es decir,

si el tipo ha sido Exacto, Completo o Complementario la explicación simplemente expone que se cumplen todos los requisitos, y si el tipo de Potencial o Parcial la explicación se genera a partir del resultado de la Abducción o Contracción y expondrá los requisitos que no se cumplen o que están en conflicto.

6. Una vez acabado el procesamiento de todas las recomendaciones sobre las ofertas, se siguen los mismos pasos pero ahora con las recomendaciones sobre las demandas. Las únicas diferencias en los pasos explicados anteriormente para el caso de las ofertas están: En el paso 1, donde ahora se buscan las entradas donde el "type_query" sea "DEMAND" y en el paso 2, donde la creación de una la base de conocimiento es definida con el contenido de las Ontologías que definen la semántica para la recomendación desde el punto de vista de la demanda.

5.2.4.2 Clases

A continuación se presentan brevemente las clases Java que componen este módulo.

5.2.4.2.1 Clase ProcessorMatch

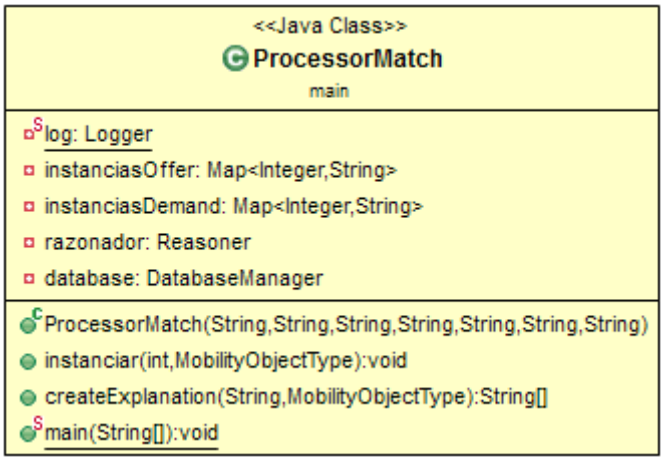


Figura 5-9: Clase ProcessorMatch

Esta es la clase principal del módulo la cual implementa el algoritmo de recomendación presentado anteriormente. Para ello se apoya en la clase "Reasoner" que realiza toda la comunicación con "MaMaS-tng".

5.2.4.2.2 Clase Reasoner

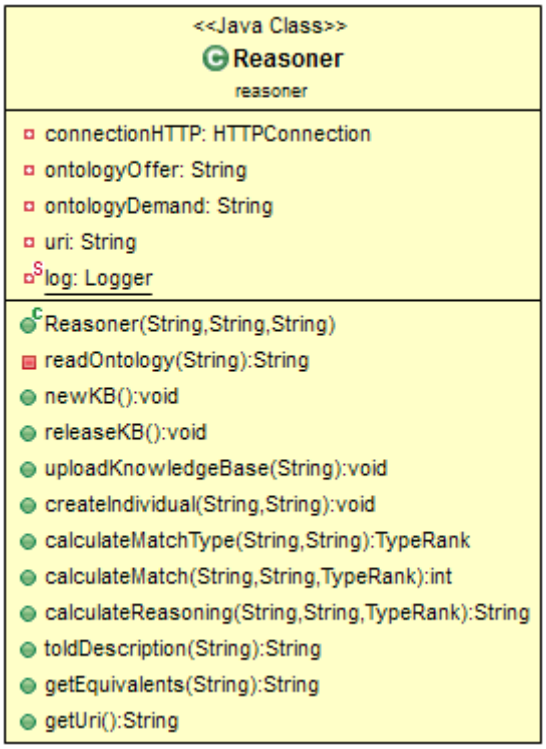


Figura 5-10: Clase Reasoner

Esta clase es la que realiza toda la comunicación con el razonador "MaMaS-tng" permitiendo así poder acceder a este de una forma sencilla por el algoritmo principal, abstrayéndole de lo referente a la comunicación. Es decir, ofrece una interfaz del razonador permitiendo realizar las peticiones o inferencias requeridas para llevar a cabo el algoritmo de recomendación. Todas estas inferencias son implementadas por sus correspondientes métodos, los cuales realizan la comunicación con "MaMaS-tng", mediante su interfaz DIG, enviando las peticiones HTTP correspondientes a las inferencias e interpretando las correspondientes respuestas HTTP.

5.2.4.2.3 Clase Petition

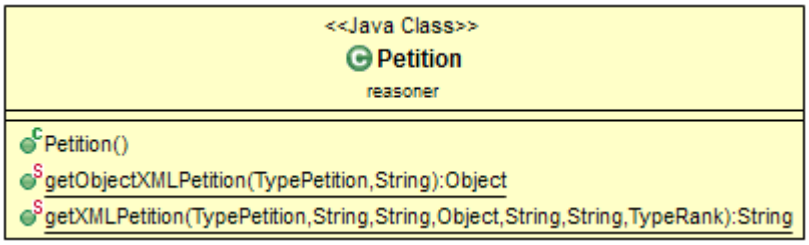


Figura 5-11: Clase Petition

Esta clase es empleada por la clase “Reasoner” para generar los mensajes DIG que se envían en las peticiones HTTP y para interpretar los mensajes DIG contenidos en las respuestas HTTP. Por un lado, la generación de mensajes se lleva a cabo mediante la serialización de objetos Java a xml. Esos objetos pertenecen a clases que representan un mensaje DIG específico. Y por otro lado, la interpretación de mensajes se lleva a cabo mediante la deserialización de xml en su correspondiente objeto Java que lo representa. De esta manera para crear los mensajes simplemente es necesario crear el objeto correspondiente el cual recibirá como parámetros la información que define a ese mensaje y para recuperar la información de un mensaje consultando los atributos de su objeto. Todo este proceso de serialización y deserialización se realiza por medio de la librería de Java JAXB y a partir de distintas clases que definen los mensajes necesarios. Estas clases no son mostradas en esta descripción ya que todas siguen el mismo patrón y únicamente se componen de atributos anotados con su correspondiente etiqueta xml.

5.2.5 Interfaz gráfica (GUI)

Como se ha venido detallando en anteriores ocasiones, el sistema es accedido a través de la red social ABP implementada con Elgg, por ello para implementar la interfaz gráfica se ha desarrollado un plugin Elgg para ser instalado en ABP. El plugin ofrece todas las interfaces necesarias para la gestión de las ofertas, demandas y recomendaciones y todas sus funcionalidades asociadas.

Para detallar la implementación de este plugin primeramente se presenta la estructura estándar, marcada por Elgg [44], de directorios y archivos que componen el plugin, la cual ha sido seguida y a continuación se describen los ficheros más relevantes de cara al funcionamiento.

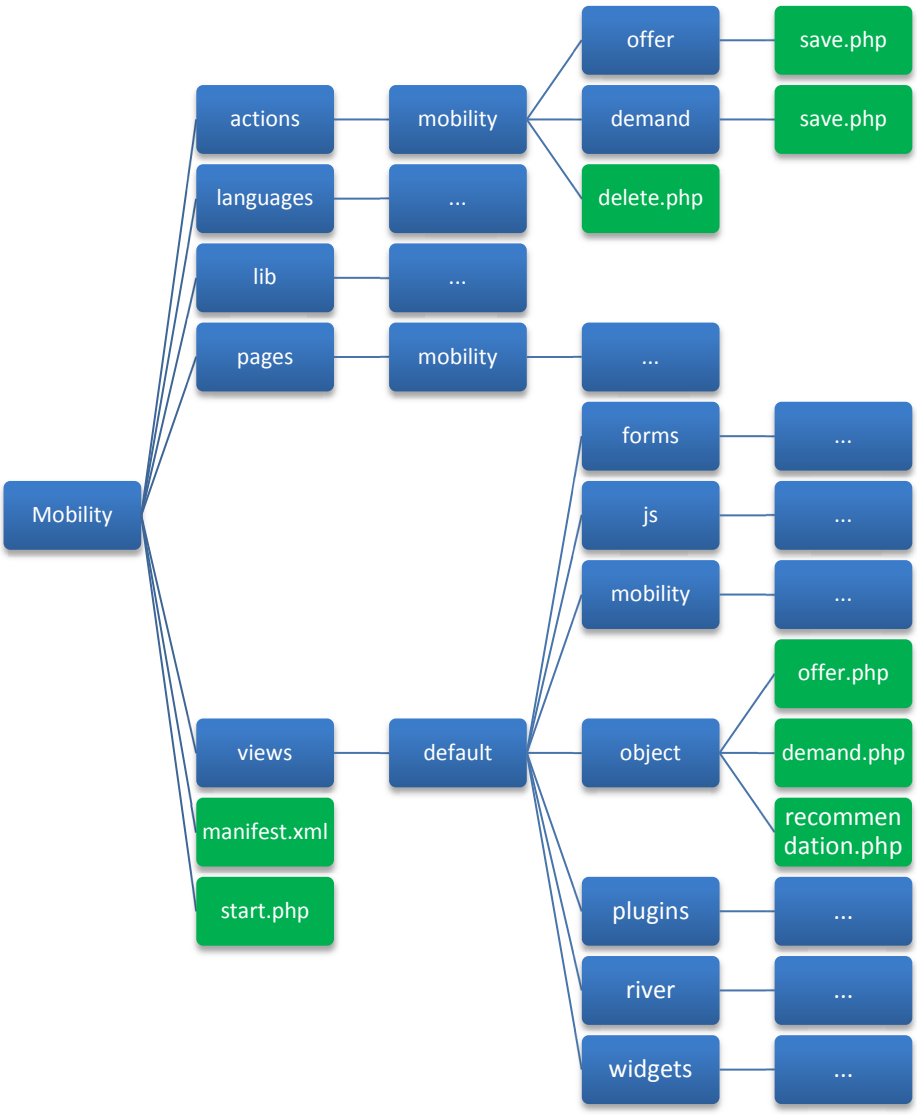


Figura 5-12: Esquema y estructura del plugin para AFRICA BUILD Portal

En la anterior figura las cajas verdes representan ficheros, las cajas azules directorios y las cajas azules con puntos suspensivos a un conjunto de ficheros y subdirectorios de menor importancia que no son presentados. De esta manera comenzando por la raíz del plugin se tienen los siguientes elementos:

- manifest.xml: Este fichero es un fichero xml estándar de configuración del plugin donde se indican parámetros como el nombre y versión del plugin entre otros y se especifican los plugins requeridos para la instalación de este.

- start.php: Este fichero es clave para el funcionamiento del plugin ya que es llamado por el core de Elgg para cargar el plugin. En este fichero se registran las acciones definidas por este plugin, el manejador de páginas, las librearías JavaScript específicas, los tipos de entidades creados en el plugin o la configuración del cron para ejecutar el modulo procesador.

- actions: En este directorio están los ficheros de las acciones que se han creado, siguiendo el esquema de nombrado y jerarquización de directorios, las acciones creadas aparecen dentro un subdirectorio que tiene como nombre el tipo de entidad. Por un lado, los ficheros save.php definen las acciones que se realizan cuando se crea o edita una oferta o demanda mediante su correspondiente formulario de creación o edición. De esta manera las acciones que se realizan al crear o editar una oferta o demanda son: Crear (editar) la entidad del tipo correspondiente con la información correspondiente, insertar o actualizar la información en la base de datos externa, insertar una entrada nueva en la actividad del sistema con la información acerca de la oferta o la demanda, y también es desde aquí desde donde se llama al ejecutable del módulo del preprocesador del Match, lo que se realiza mediante una llamada a la función "exec()" de php en el que se indica el comando Windows o Linux para ejecutar el fichero preprocessor.jar. A continuación se muestra el ejemplo de la creación de una entidad de tipo "demanda".

```
52 $demand = new ElggObject();
53 $demand->access_id = 2;
54 
56 $demand->title = $title;
57 $demand->description = $description;
58 $demand->tags = string_to_tag_array($tags);
59 $demand->save();
```

Por otro lado, dentro de las acciones también está el fichero delete.php. En este fichero se definen las acciones a realizar al eliminar una oferta o demanda, es decir, se borra la entidad asociada a esa oferta o demanda, se elimina la entrada de la base de datos externa correspondiente a esa oferta o demanda y se elimina la entrada de la actividad del sistema.

- languages: En este directorio se encuentran los archivos que soportan la traducción de los textos mostrados, a los diferentes idiomas soportados por ABP. Estos ficheros son ar.php, en.php, es.php y fr.php que definen los textos para el árabe, inglés, español y francés. Cada fichero declara un array de pares clave-valor, los valores son los diferentes fragmentos de texto que son mostrados en las páginas del mobility y las claves identifican a estos textos, teniendo en los tres ficheros las mismas claves pero con valores distintos correspondientes a la traducción específica de ese texto. Estas claves son empleadas en las vistas o páginas mediante la función elgg_echo('clave') la cual imprime el texto correspondiente a esa clave según el idioma que tenga configurado el usuario.

- lib: En este directorio se encuentran librerías que son empleadas por el plugin, en concreto se tienen los ejecutables de los módulos del preprocesador del Match y del procesador del Match, es decir, los ficheros preprocessor.jar y processor.jar, además de tres ejecutables correspondientes a la librería OpenNLP que emplea el modulo del preprocesador.

- pages: En este directorio están los archivos ".php" que generan las distintas páginas HTML que conforman la interfaz gráfica de usuario, es decir, la página donde se le muestran las ofertas y demandas, la página donde visualiza una oferta o demanda, la página donde visualiza sus recomendaciones, etc. Todas estas páginas esta implementadas siguiendo los esquemas de Elgg que marca hacer uso de views que sean o bien predefinidas por Elgg las cuales crean elementos como menús, botones, encabezados, etc. o bien definidas por el usuario para sus propios objetos. En total para la implementación de toda la interfaz de usuario se han creado 13 páginas, a continuación se muestra el resultado de las más representativas.

  La siguiente imagen muestra la página principal del mobility. Esta página se puede dividir en tres secciones, la primera que muestra la cabecera del mobility, donde aparece el camino de navegación actual "Mobility Lite > Carlos Muiño > Colleagues", el título de la página actual "Colleagues' offers and demands" y los botones "Create an offer" y "Create a demand"; la segunda sección, es el sidebar del extremo derecho, el cual es mostrado en todas las páginas del mobility. En

este sidebar se encuentra el botón "Recommendations" para acceder a las recomendaciones y un formulario para realizar búsquedas; la tercera sección es la parte principal o el cuerpo de la página, este cuerpo se compone por dos pestañas distintas, la pestaña "Colleagues" que muestra las ofertas y demandas pertenecientes a los amigos del usuario y la pestaña "Mine" que muestra las ofertas y demandas del propio usuario.

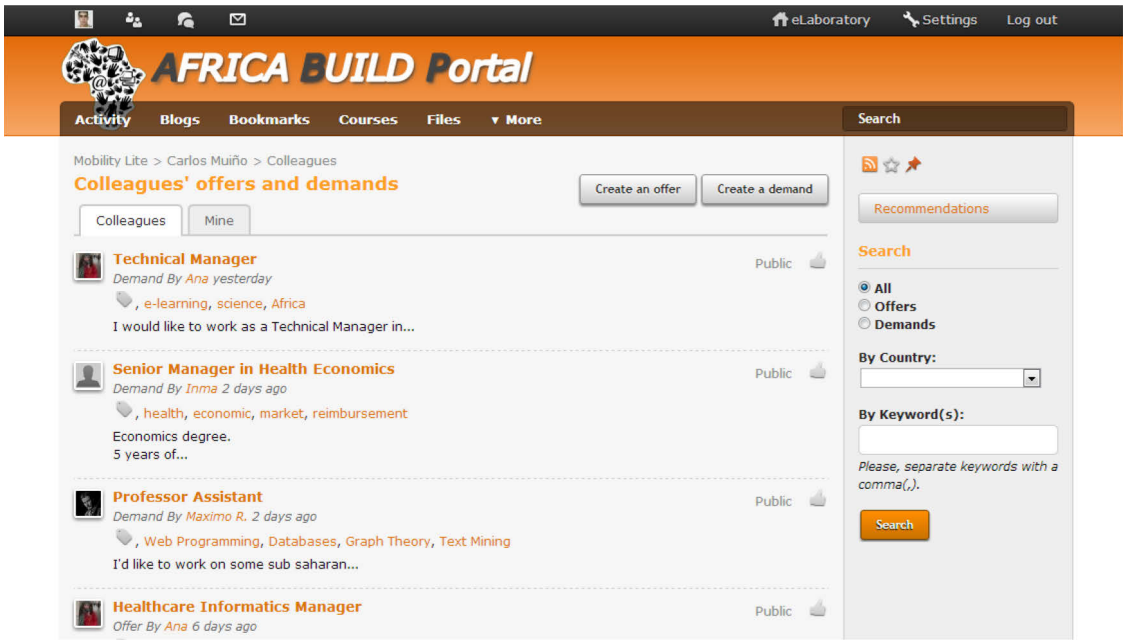


Figura 5-13: Página principal del Mobility en AFRICA BUILD Portal

La siguiente imagen muestra la pantalla donde el usuario visualiza sus recomendaciones. Esta página sigue la misma estructura que todas las páginas del Mobility dividiéndose en cabecera, sidebar y sección principal. En esta sección principal se muestran todas sus recomendaciones diferenciándolas entre las que son acerca de sus demandas y acerca de sus ofertas. Cada recomendación es un elemento desplegable con el título de su demanda u oferta y en su contenido desplegado contiene la lista de las ofertas y demandas recomendadas, respectivamente. Cada elemento recomendado presente en esta lista muestra el título del elemento, su proximidad o ranking y un indicador de cuál es el principal requisito no cumplido, el cual es un enlace a la página donde se presenta la explicación detallada de la recomendación.

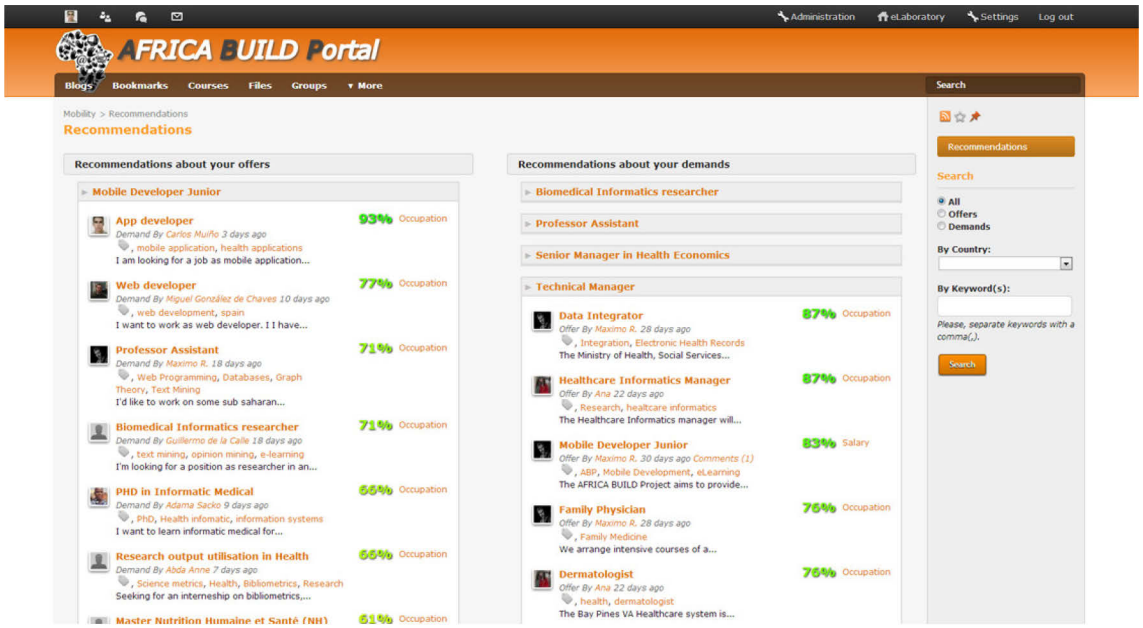


Figura 5-14: Página de visualización de las recomendaciones en AFRICA BUILD Portal

- views: En este directorio se encuentran todas las views necesarias para generar los contenidos de las páginas presentadas anteriormente. Estas views son plantillas para generar ciertos elementos de manera que para mostrar un mismo elemento en diversos sitios no se tenga que volver a escribir el mismo código HTML sino que simplemente basta con llamar a la función elgg_view('view') indicándole el nombre de la view a mostrar y una seria de parámetros o argumentos.

  Dentro del directorio views se encuentra el directorio default, que indica que estas son las vistas para mostrar por defecto, esto es cuando el sitio se visualiza a través de una navegador web de ordenador. Dentro de este directorio se tienen otros directorios. El primero de ellos, forms, contiene las vistas de los formularios de creación y edición de las ofertas y demandas, los cuales son formularios tradicionales con campos de texto, elementos seleccionables, listas despegables, etc. El siguiente directorio, js, contiene librerías JavaScript y jQuery empleadas en el plugin, como por ejemplo en los formularios para validar que se han completado los campos requeridos. El siguiente directorio mobility, contiene varias vistas empleadas por el plugin como por ejemplo un sidebar para realizar las búsquedas, etc. El siguiente directorio, object, es el más destacado ya que contiene las vistas asociadas a las entidades definidas por el plugin, las entidades tipo offer y demand. Los ficheros offer.php, demand.php

definen estas vistas, y siguiendo el esquema de Elgg, en esa misma vista incluyen diferentes modos de presentación de la vista, esto es, por ejemplo se tienen 3 distintas representaciones de la oferta y la demanda, una de ellas es cuando se muestran completas, mostrando todos sus campos e información, otra de ellas es cuando se muestran como elementos de una lista de ofertas y demandas, usada por ejemplo en la página principal del mobility y otra de ellas es la usada cuando se muestran en los Widgets. Además de estas dos vistas también se tiene la vista de la recomendación que aunque no es una entidad Elgg, es un objeto más con propia representación gráfica. El siguiente directorio, plugins, contiene la vista que se integra en el panel general de administración de ABP, la cual sirve para configurar la parte asociada a este plugin. El siguiente directorio, river, contiene las vistas de las entradas que se crean en la actividad del sistema cuando se crea una oferta o demanda, estos elementos siguen el esquema de Elgg para ello muestra el título de objeto creado, el autor de la creación y una breve descripción del objeto. El último directorio, widgets, contiene las vistas que se encargan de generar los widgets que tiene el sistema, que son tres, uno para mostrar las ultimas demandas registradas, otro para mostrar las últimas ofertas registradas y el último para mostrar las últimas recomendaciones de las ofertas y demandas del usuario. Estos plugins siguen la estructura de Elgg, ofreciendo parámetros de configuración.

La siguiente imagen muestra la página de visualización de una demanda, la cual es generada mediante la vista asociada a la entidad de tipo “demand”.

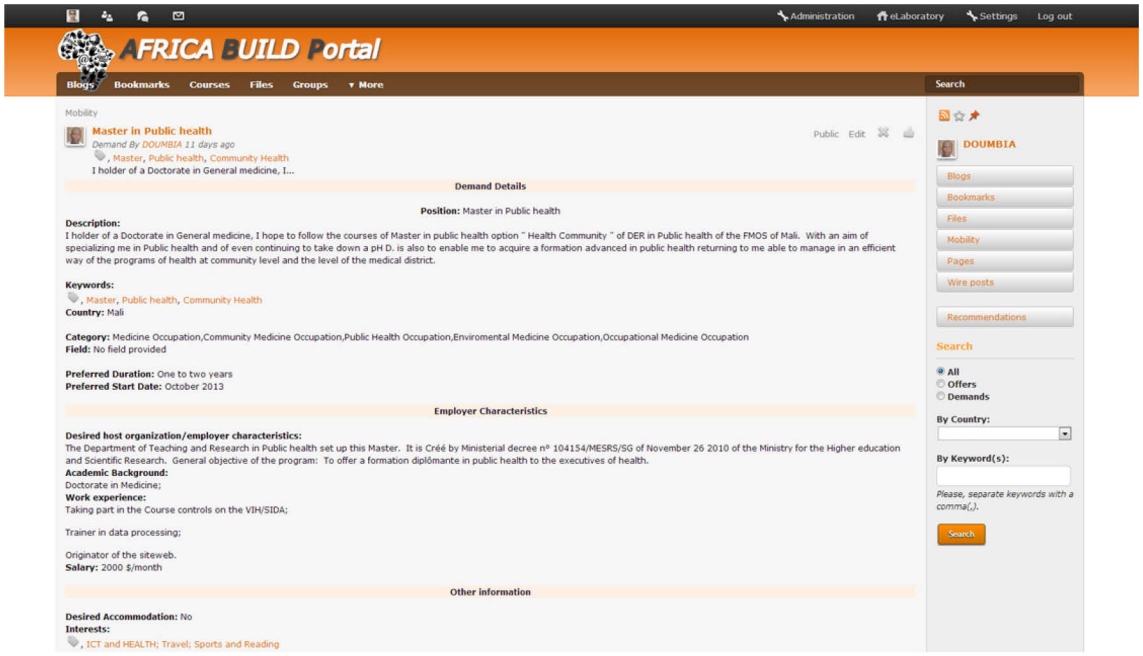


Figura 5-15: Vista de una demanda en AFRICA BUILD Portal

Capítulo 6

PRUEBAS Y RESULTADOS

6.1 Pruebas del sistema

En esta sección se presentan las pruebas que se han realizado para evaluar y certificar el correcto funcionamiento del sistema.

Antes de integrar el sistema en su entorno real, es decir, en la versión en producción de AFRICA BUILD Portal [45] se han realizado diversos tipos de pruebas para corregir errores de implementación. Estas pruebas han sido: pruebas unitarias para probar la interfaz gráfica y los módulos procesador y preprocesador; pruebas de integración para probar la integración de la interfaz gráfica con el preprocesador y con el procesador; y pruebas del sistema para validar los objetivos y requisitos del proyecto. En este apartado no se presentan ni la descripción ni los resultados de las pruebas unitarias y de integración ya que se da mayor relevancia a las pruebas del sistema.

El objetivo del plan de pruebas, presentado a continuación, es valorar si las recomendaciones efectuadas por el sistema son válidas. Esta validez se refiere a que las recomendaciones se asemejen a las que un usuario real efectuaría en base a su razonamiento.

El plan de pruebas del sistema ha consistido en realizar una comparación entre las recomendaciones efectuadas por el sistema y las efectuadas de manera manual por usuarios reales, sobre un mismo conjunto de datos. El proceso seguido para la ejecución del plan de pruebas ha sido el siguiente:

1. El primer paso ha sido generar las ofertas para el conjunto de datos que será empleado para realizar las recomendaciones. Para no elaborar un conjunto de datos condicionado por el conocimiento acerca del desarrollo y funcionamiento del sistema, estas ofertas han sido elaboradas por personas ajenas al desarrollo del sistema. En concreto se ha contado con la colaboración de miembros de otras instituciones del proyecto AFRICA BUILD, a los cuales se les pidió que crearan una serie de ofertas inventadas pero ajustadas a posibles necesidades que tuvieran en sus instituciones. El número de total de ofertas elaboradas ha sido 18, cifra que se estimó para cubrir todos los posibles casos de error.

2. El segundo paso, una vez creadas todas las ofertas, ha sido generar las demandas para el conjunto de datos. De igual manera que con las ofertas, las demandas han sido elaboradas por miembros de otras instituciones de AFRICA BUILD, pero no por los mismos que crearon las demandas, para tampoco estar condicionadas estas demandas. En este caso se les pidió que crearan demandas imaginarias pero basadas en lo que ellos buscarían en función a sus preferencias, conocimientos, etc. Al mismo tiempo a cada usuario se le pidió que realizara cinco recomendaciones para su demanda. Estas recomendaciones se refieren a que el usuario seleccionara y ordenara las cinco mejores ofertas en las cuales el usuario estaría interesado, de entre las 18 ofertas que fueron creadas anteriormente.

3. El tercer paso ha sido ejecutar el sistema para que efectuara todas las recomendaciones entre las ofertas y demandas creadas en los pasos anteriores.

4. El último paso del plan de pruebas ha sido contrastar las recomendaciones, creadas por las dos fuentes, aplicando medidas de evaluación del rendimiento para sistemas de recuperación de información.

En el siguiente apartado se presenta la definición de estas medidas de evaluación y los resultados de estas pruebas basados en estas medidas.

6.2 Resultados

Las medidas para evaluar el rendimiento de un sistema de recuperación de información están basadas en conjuntos, es decir, requieren de una colección de documentos y de una consulta al sistema. En este caso como consulta tenemos la demanda de cada usuario y como documentos las ofertas recomendadas por el sistema a esa demanda.

Las medidas de evaluación que se han empleado para evaluar el sistema, han sido la precisión y la exhaustividad, las cuales son relacionadas resultando la curva de precisión-exhaustividad. Donde, la exhaustividad indica la capacidad del sistema para presentar todos los elementos relevantes y la precisión indica la capacidad del sistema para presentar solamente los elementos relevantes. Siendo sus definiciones matemáticas:

Exhaustividad = *Nº elementos relevantes recuperados / Nº elementos relevantes en la colección*.

Precisión = *Nº elementos relevantes recuperados / Nº total de elementos recuperados*.

El cálculo de esta curva de precisión-exhaustividad se realiza marcando como relevante o no relevante cada oferta recomendada por el sistema a cada demanda del usuario, donde relevante o no relevante indica si la oferta marcada se encuentra entre las seleccionadas por el usuario de manera manual o no. De esta manera se ha realizado el cálculo de una curva por cada demanda, obteniendo 12 curvas distintas las cuales son presentadas en el Anexo II. Posteriormente, se ha calculado la curva de precisión-exhaustividad media del sistema a partir de la interpolación lineal de las 12 curvas anteriores.

La siguiente tabla muestra los valores de esta curva resultante para los valores de los niveles estándar de exhaustividad (0 a 1 en incrementos de 0,1).

Exhaustividad	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
Precisión	0,70	0,70	0,70	0,55	0,55	0,48	0,48	0,42	0,42	0,36	0,36

Tabla 9: Valores de la curva precisión-exhaustividad del sistema

La siguiente grafica representa los valores de la tabla anterior.

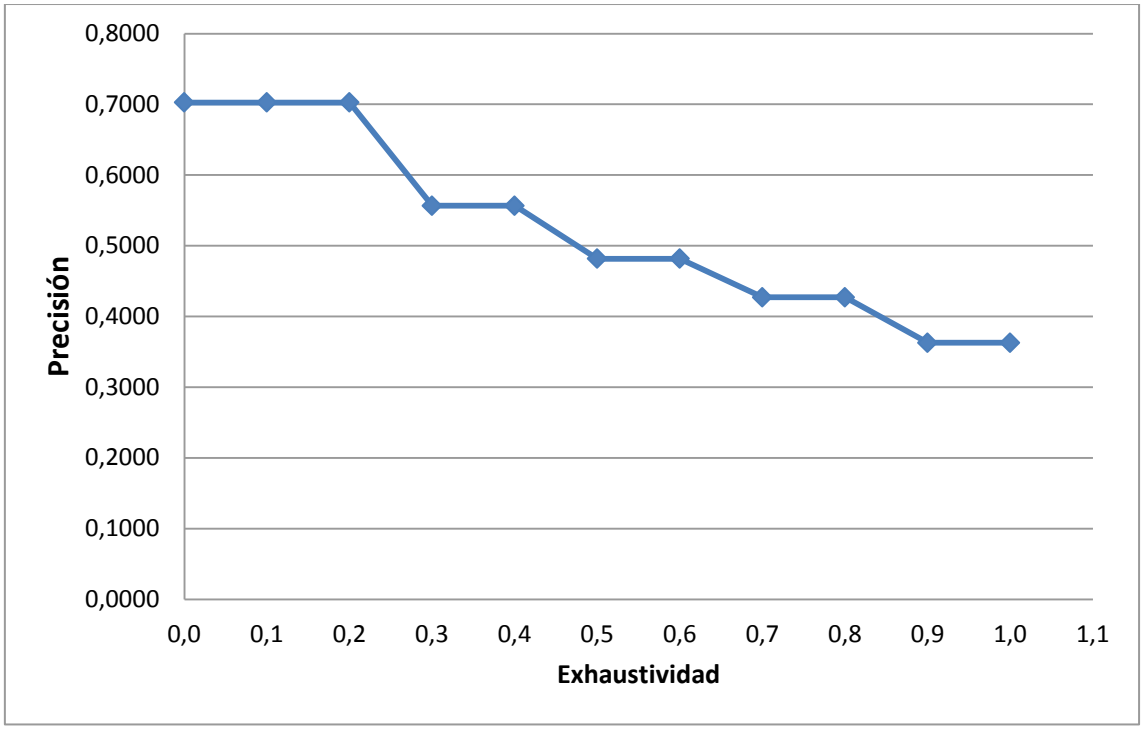


Figura 6-1: Gráfica de la curva precisión-exhaustividad del sistema

A la vista de la curva resultante del rendimiento del sistema se pueden realizar las siguientes afirmaciones:

- La curva presenta una pendiente descendente de izquierda a derecha, lo cual ocurre, típicamente, en estos sistemas de recuperación de información.

- El objetivo de estos sistemas es que presenten curvas donde la exhaustividad y precisión se encuentren maximizadas, es decir, la curva este lo más cerca posible de la esquina superior derecha. Este objetivo es difícil de lograr en su completitud, por lo que se busca que la precisión y exhaustividad sea lo mayor posible. En este caso se puede afirmar que se llega a la exhaustividad máxima (1,0) presentando una precisión del 36%. Lo que indica que de cada 100 ofertas recomendadas 36 de ellas son las que un usuario consideraría relevantes.

- La exhaustividad máxima indica, en este caso, que todas las ofertas que el usuario determina como relevantes han sido recomendadas por el sistema. En los sistemas de recuperación de información no siempre tiene porque alcanzarse tal nivel. En este sistema si se alcanza debido a que se generan todos los posibles pares de recomendación entre ofertas y demandas, aunque en la mayoría de los casos estas tendrán una proximidad muy baja.

- La precisión máxima (100%) nunca se llega a alcanzar, incluso ni en niveles bajos de exhaustividad, sin embargo se parte de una precisión aceptable (70%) y su precisión más baja es de un 36%.

Las discrepancias en la comparación de las recomendaciones las cuales han generado estos resultados son fruto de defectos o carencias del sistema. Estos defectos han sido detectados y su aparición se puede deber a uno de estos tres aspectos:

- Expresividad del sub-lenguaje de Lógica Descriptiva requerido por "MaMaS-tng". "MaMaS-tng" cuenta con un lenguaje de Lógica Descriptiva ALN, lo que provoca algunas limitaciones para representar la información de las ofertas y demandas. Las principales limitaciones se encuentran en la imposibilidad de anidar roles y en la inexistencia del constructor de la disyunción. Esto provoca que ciertas formalizaciones de las ofertas y demandas no signifiquen semánticamente lo mismo que la información que se describe en esas ofertas y demandas. Por ejemplo, no es posible relacionar cada rol de conocimiento con su correspondiente rol de nivel de conocimiento o el hecho de indicar varios países en una demanda supone que se demanda un trabajo en todos esos países.

- Alcance y definición de las Ontologías. Las ontologías implementadas pueden no abarcar todo el dominio y por otro lado la semántica que definen puede no corresponderse con las interpretaciones de los usuarios, en algunos casos.

- Formalización de las ofertas y demandas a partir de su descripción en lenguaje natural. Dado que se realiza la formalización de campos que se encuentran expresados en lenguaje natural, al igual que en el primer tipo de defectos, su

formalización puede no corresponderse a lo expresado. Esto se debe a que ciertas representaciones gramaticales no estén soportadas, no se detecten ciertos conceptos mediante la sinonimia o se produzcan interpretaciones erróneas del significado de los verbos.

Capítulo 7

CONCLUSIONES Y LÍNEAS FUTURAS

7.1 Conclusiones

El objetivo general planteado en este Trabajo de Fin de Carrera era diseñar e implementar un sistema de recomendación semántico que estuviera incluido en el entorno de AFRICA BUILD Portal.

La solución propuesta basada en la combinación de distintas tecnologías y servicios como un razonador de lógica descriptiva, tecnologías de procesamiento de lenguaje natural y tecnologías Web, ofrece las funciones requeridas y planteadas en los objetivos del sistema.

El sistema permite la generación de las recomendaciones de una forma automática y transparente al usuario. Presentando estas recomendaciones de una manera clasificada y proporcionando una explicación en lenguaje natural a la recomendación.

El sistema implementado es un sistema sólido y modular, posibilitando así su fácil evolución. Teniendo como base de su evolución las ontologías, las cuales pueden ser modificadas para expandir su dominio e incluir nueva semántica para el proceso de recomendación.

El enfoque modular del sistema ha propiciado la optimización e integración con AFRICA BUILD Portal, realizándose las tareas de procesamiento de las recomendaciones por módulos independientes al portal, pero por otro lado logrando una integración con él de una manera totalmente transparente de cara al usuario.

Con los resultados obtenidos en la evaluación del sistema se puede concluir que aunque los resultados obtenidos no sean óptimos, estando ante un sistema semántico se puede considerar que el sistema es efectivo y satisfactorio. Un segundo ciclo de desarrollo, requerido en todos los sistemas de ese tipo, implicaría la refinación de los módulos sujetos a error, como el preprocesador y las ontologías para maximizar el área de la curva precisión-exhaustividad y por tanto mejorar la calidad de las recomendaciones ofrecidas. Dichos ajustes se realizarían en los aspectos sujetos a error descritos anteriormente y en base a los criterios aportados por los usuarios.

7.2 Líneas futuras

A pesar de que los objetivos de este Trabajo Fin de Grado han sido alcanzados y de que se han obtenido resultados satisfactorios de acuerdo a esos objetivos, como se ha expuesto en el capítulo anterior, es posible evolucionar el sistema en dos sentidos: Por un lado para corregir ciertas limitaciones que presenta el sistema y por otro lado para aumentar su funcionalidad. Por ello en este apartado se presentan posibles líneas futuras que podrían ser desarrolladas indicando su prioridad y complejidad.

- Traducción de las ontologías.
  Prioridad: Alta, Complejidad: Baja.
  Ya que actualmente AFRICA BUILD Portal da soporte a los idiomas árabe, español, francés e inglés, se debería permitir la redacción de las ofertas y demandas en estos idiomas así como ofrecer las explicaciones a las recomendaciones en esos mismos idiomas. Para ello sería necesaria la traducción de estas ontologías a cada uno de estos idiomas.

- Refinamiento del sistema para maximizar su rendimiento.
  Prioridad: Alta, Complejidad: Media.
  Con el objetivo de maximizar el área de la curva precisión-exhaustividad una segunda fase de desarrollo debería refinar el sistema. Por un lado este refinamiento se debería hacer sobre las ontologías para ajustar su semántica a las de los usuarios, en aquellos puntos en lo que presentan diferencias. Por ejemplo las mayores discrepancias detectadas están en el contraste del salario ya que con la solución actual una diferencia en el salario penaliza demasiado. Por otro lado, también se podría mejorar el modulo preprocesador para extraer información que ahora no se extrae o que se extrae incorrectamente.

- Explotación del potencial de AFRICA BUILD Portal mediante la Web 3.0.
  Prioridad: Media, Complejidad: Media.
  AFRICA BUILD Portal es una potente red social que dispone de diversas herramientas y posibilidades que los usuarios emplean. Este uso de la red genera información de gran valor acerca de los usuarios. Así, determinada información de este tipo podría ser incluida y empleada para realizar las recomendaciones entre las ofertas y demandas. En concreto, a día de hoy, se podrían incluir como conocimientos académicos del demandante los conocimientos que se consideran obtenidos al realizar y superar un curso académico mediante el módulo de e-learning de ABP [46].

- Maximización de las recomendaciones.
  Prioridad: Media, Complejidad: Alta.
  Además de la inclusión de información derivada del uso de la red social, en el proceso de recomendación se podría incluir más información de las partes a contrastar. Esta información a incluir podría ser de elementos que actualmente definen a las ofertas y las demandas, pero que no son tenidos en cuenta al realizar las recomendaciones. En concreto se podría contrastar información acerca de la experiencia, descripciones o diversos requisitos.

  Esta inclusión provocaría actualizar las ontologías para dar soporte al razonamiento sobre esa información y modificar el módulo preprocesador para generar la formalización de esa nueva información.

- Implementación de un razonador de Lógica Descriptiva.
  Prioridad: Media, Complejidad: Alta.
  Ante las carencias o limitaciones de "MaMaS-tng" evidenciadas en el capítulo anterior se podría implementar un razonador de Lógica Descriptiva propio. Este razonador debería presentar las mismas características y cualidades que "MaMaS-tng", es decir, ofrecer sus mismas inferencias y disponer al menos de una interfaz DIG, pero además debería contar con un sub-lenguaje de Lógica Descriptiva más descriptivo para suplir las limitaciones de expresividad.

- Inclusión de un espacio de negociación.
  Prioridad: Baja, Complejidad: Baja.
  La interfaz provista por el plugin para ABP podría incluir un apartado de negociación de la recomendación. Esta negociación supondría dar al usuario la posibilidad de marcar las recomendaciones, efectuadas a sus ofertas y demandas, como válidas o no válidas y de realizar un posterior seguimiento. De esta manera, por un lado las marcadas como no validas no le volverían a aparecer ni serian calculadas de nuevo y por otro lado con las marcadas como válidas se podría realizar una negociación a partir de la redefinición de esas ofertas y demandas y de la comunicación entre ofertante y demandante a través de ABP.

Capítulo 8

BIBLIOGRAFÍA

[1] Ana Jiménez Castelllanos, Diana de la Iglesia, Máximo Ramírez Robles, and Victor Maojo, "The AFRICA BUILD Project: Building a research and educational infrastructure for health in Africa" in *Open Access Africa*, Kumasi, 2011.

[2] Ana Jiménez Castellanos, Máximo Ramírez Robles, Diana de la Iglesia, and Víctor Maojo, "Building sustainable capacity for research for health in Africa: first stages of the AFRICA BUILD Project" in *VPH 2012*, London, 2012.

[3] The Evolution of the Web. [Online]. http://www.evolutionoftheweb.com/, accedido por última vez en Marzo de 2013.

[4] George Aspridis, Vasiliki Kazantzi, and Dimitrios Kyriakou, "Social Networking Websites and Their Effect in Contemporary Human Resource Management" in *Mediterranean Journal of Social Sciences*, 2013.

[5] Guillermo De la Calle, Victor Maojo, and Mario Benito, "The INFOBIOMED Network of Excellence: Facilitating Training and Mobility in Biomedical Informatics in Europe" *Stud Health Technol Inform*, 2006.

[6] Hyacinth Sama Nwana, "Software Agents: An Overview," in *Knowledge Engineering Review*: Cambridge University Press, 2009.

[7] Wikipedia. Matchmaking. [Online]. http://en.wikipedia.org/wiki/Matchmaking, accedido por última vez en Marzo de 2013.

[8] WordReference. [Online]. www.wordreference.com, accedido por última vez en Abril de 2013.

[9] Daniel Kuokka and Larry Harada, "Matchmaking for Information Agents" in *Readings in Agents*: Morgan Kaufman, 1995.

[10] Daniel Veit, Jorg P. Muller, Martin Schneider, and Bjorn Fienh, "Matchmaking for Structured Objects" in *Third International Conference on Data Warehousing and*

Knowledge Discovery, DaWaK, Munich, 2001.

[11] Daniel Veit, Jorg P. Muller, Martin Schneider, and Bjorn Fienh, "Matchmaking for Autonomous Agents in Electronic Marketplaces" in *Fith International Conference on Autonomous Agents*, Montreal, 2001.

[12] Tommaso Di Noia, Eugenio Di Sciascio, and Francesco M. Donini, "Semantic Matchmaking as Non-Monotonic Reasoning: A Description Logic Approach" *Journal of Artificial Intelligence Research (JAIR)*, 2007.

[13] Tommaso Di Noia, Eugenio Di Sciascio, Francesco M. Donini, and Marina Mongiello, "A system for Principled Matchmaking in an Electronic Marketplace" *International Journal of Electronic Commerce*, 2004.

[14] Daniel J. Veit, *Matchmaking in Electronic Markets - An Agent-Based Approach towards Matchmaking in Electronic Negotiations*: Springer, 2004.

[15] Simona Colucci, Tommaso Di Noia, Eugenio Di Sciascio, Francesco M. Donini, and Marina Mongiello, "Concept abduction and contraction for semantic-based discovery of matches and negotiation spaces in an e-marketplace" *Electronic Commerce Research and Applications*, 2005.

[16] Franz Baader, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, *The Description Logic Handbook*. Cambridge: Cambridge University Press, 2010.

[17] DESCRIPTION LOGIC REASONERS. [Online]. http://www.cs.man.ac.uk/~sattler/reasoners.html, accedido por última vez en Abril de 2013.

[18] Steffen Staab and Rudi Studer, *Handbook on Ontologies*: Springer, 2009.

[19] Thomas R. Gruber, "A Translation Approach to Portable Ontology Specifications" *Knowledge Acquisition - Current issues in Knowledge modeling*, 1993.

[20] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho, *Ontological Engineering*: Springer, 2004.

[21] W3C. Standards, Semantic Web. [Online]. http://www.w3.org/standards/semanticweb/ontology, accedido por última en vez Mayo de 2013.

[22] Jon Duckett, *Beginning Web Programming with HTML, XHTML, and CSS*: Wrox, 2004.

[23] W3C. Standards, Web Design and Applications. [Online]. http://www.w3.org/standards/webdesign, accedido por última vez en Abril de 2013).

[24] W3C. Guía Breve de CSS. [Online]. http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo, accedido por última vez en Abril de 2013.

[25] The jQuery Foundation. jQuery. [Online]. http://jquery.com/, accedido por última vez en Abril de 2013.

[26] The PHP Group. php. [Online]. http://php.net/, accedido por última vez en Abril de 2013.

[27] Cash Costelo, *Elgg 1.8 Social Networking:* Packt Publishing, 2012.

[28] W3C. OWL Web Ontology Language - Overview. [Online]. http://www.w3.org/TR/owl-features/, accedido por última vez en Abril de 2013.

[29] W3C. RDF Primer. [Online]. http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, accedido por última vez en Mayo de 2013.

[30] W3C. SPARQL Query Language for RDF. [Online]. http://www.w3.org/TR/rdf-sparql-query/, accedido por última vez en Mayo de 2013.

[31] The Apache Software Foundation. Apache Jena. [Online]. http://jena.apache.org/, accedido por última vez en Mayo de 2013.

[32] Sean Bechhofer, "The DIG Description Logic Interface: DIG/1.1" University of Manchester, 2003.

[33] DIG 1.1 XMLBeans API. [Online]. http://dig.sourceforge.net/, accedido por última vez en Mayo de 2013.

[34] Stefano Coppi, Francesco di Cugno, and Eufemia Tinelli, "Extended MaMaS DIG Description Logic Interface 1.1".

[35] James Gosling, Bill Joy, Guy L. Steele, and Gilad Bracha, *The Java Language Specification*, Tercera edición ed: ADDISON-WESLEY, 2005.

[36] Oracle. The Java Programming Language and the Java Platform. [Online]. http://www.oracle.com/technetwork/topics/newtojava/downloads/index.html, accedido por última vez en Mayo de 2013.

[37] MySQL. MySQL 5.6 Reference Manual. [Online]. http://dev.mysql.com/doc/refman/5.6/en/index.html, accedido por última vez en de Mayo 2013.

[38] Connector/J (JDBC) Reference. [Online]. http://dev.mysql.com/doc/refman/5.6/en/connector-j-reference.html, accedido por última vez en Mayo de 2013.

[39] The Apache Software Foundation. Apache OpenNLP. [Online]. http://opennlp.apache.org/, accedido por última vez en Mayo de 2013.

[40] WordNet. WordNet - A lexical database for English. [Online]. http://wordnet.princeton.edu/, accedido por última vez Mayo 2013.

[41] Brett Spell. JAWS - Java API for WordNet Searching. [Online]. http://lyle.smu.edu/~tspell/jaws/index.html, accedido por última vez en Mayo de 2013.

[42] Google. Google Spell Checker. [Online]. https://support.google.com/websearch/answer/1723?hl=en, accedido por última vez en Mayo de 2013.

[43] Google API Spelling Java. [Online]. https://code.google.com/p/google-api-spelling-java/, accedido por última vez en Mayo de 2013.

[44] Elgg Foundation project. Plugin skeleton. [Online]. http://docs.elgg.org/wiki/Plugin_skeleton, accedido por última vez en Junio de 2013.

[45] AFRICA BUILD Portal, social network. [Online]. http://ochoa.dia.fi.upm.es/africabuildportal/, accedido por última vez en Junio de 2013.

[46] Miguel González de Chaves Abreu, "Sistema de e-learning para el entorno de AFRICA BUILD Portal," Universidad Politécnica de Madrid, Trabajo Fin de Grado 2013.

Capítulo 9

ANEXOS

Anexo I. Comparación razonadores de Lógica Descriptiva

Razonador	Expresividad del lenguaje	Algoritmo de Razonamiento	Servicios de razonamiento	Tiempo ejecución	Interfaz
FaCT++	$ALCHOIQ_{(D)}$	Tablas	Satisfacibilidad Subsunción Clasificación	Exponencial	DIG, OWL API
HermiT	$ALCROIQ_{(D)}$	Hiper-Tablas	Satisfacibilidad Subsunción	Exponencial	OWL API
Pellet	$ALCHOIN_{(D)}$	Tablas	Satisfacibilidad Subsunción Clasificación Consultas SPARQL	Exponencial	DIG, OWL API
RacerPro	$ALCIQ_{(D-)}$	Tablas	Satisfacibilidad Subsunción Clasificación Consultas SPARQL	Exponencial	DIG, OWL API
MaMaS-tng	ALCN	Tablas	Satisfacibilidad Subsunción Abducción Contracción Clasificación potencial Clasificación parcial	Polinomial	DIG

A continuación se definen algunas de las características empleadas en la tabla comparativa.

- La expresividad del lenguaje indica el sub-lenguaje de lógica descriptiva que soporta el razonador. Cada sub-lenguaje presenta una capacidad distinta para representar el conocimiento. Así, la capacidad del sub-lenguaje se indica mediante la sucesión de una serie de caracteres, donde cada carácter indica los elementos que incluye:

  - AL se refiere al lenguaje base que cuenta con conceptos atómicos, concepto universal, concepto vacío, negación atómica, intersección, cuantificador universal y cuantificador existencial limitado al concepto universal.

  - C incluye las negaciones de conceptos arbitrarios.

  - N incluye restricciones numéricas mediante las restricciones at-least($\geq$) y at-most($\leq$).

  - I incluye roles inversos.

  - Q incluye restricciones numéricas cualificadas.

  - H incluye la jerarquización de roles.

  - O incluye un alfabeto para los objetos del dominio.

  - R incluye axiomas con inclusión de roles complejos, reflexividad y disyunción de roles.

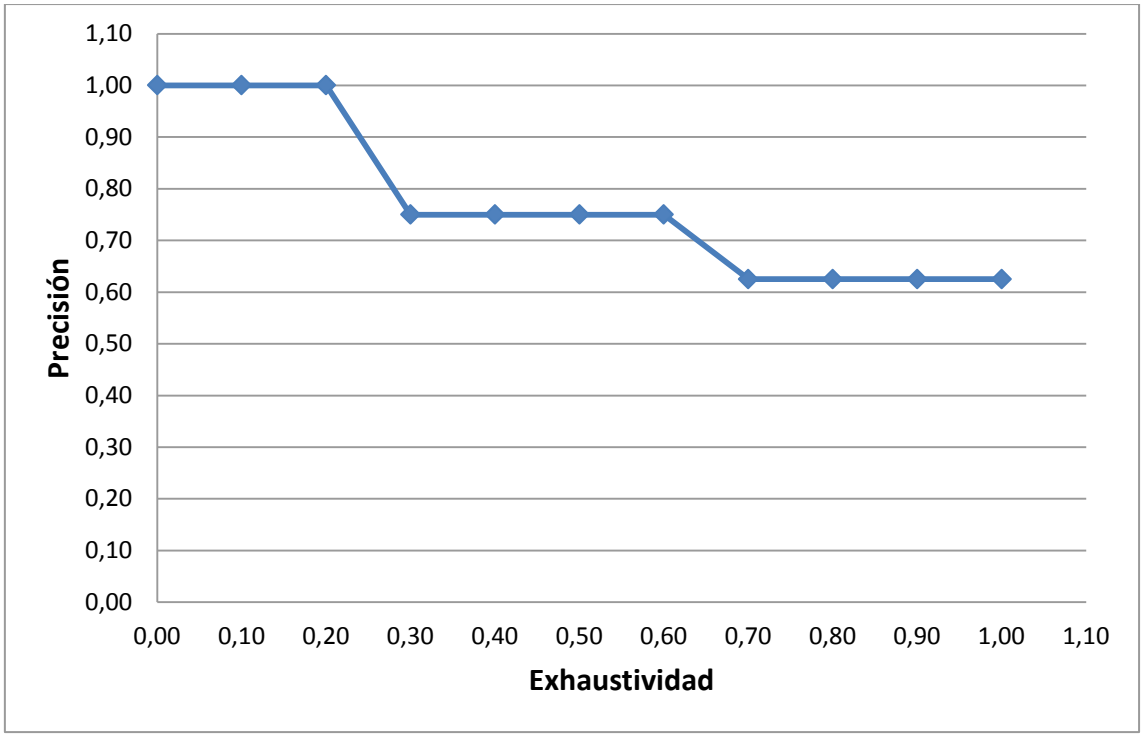
- El algoritmo de razonamiento indica el tipo de algoritmo o procedimiento de decisión mediante el cual trabaja el razonador. En los casos anteriores todos se basan en el método de tablas. Este procedimiento tiene como objetivo solucionar el problema de la satisfacibilidad. Para ello se basa en ir descomponiendo las sentencias lógicas en partes más simples y pequeñas para detectar conflictos entre ellas. Cuando no se pueden aplicar más reglas de simplificación y no se han encontrado conflictos, la sentencia es considerada satisfacible.

- Las interfaces Interfaz, permiten el acceso a los razonadores para hacer uso de sus inferencias desde distintos lenguajes de programación o mediante distintos métodos de comunicación. Las interfaces más comúnmente empleadas o soportadas son DIG, OWL-API y Jena Interface.

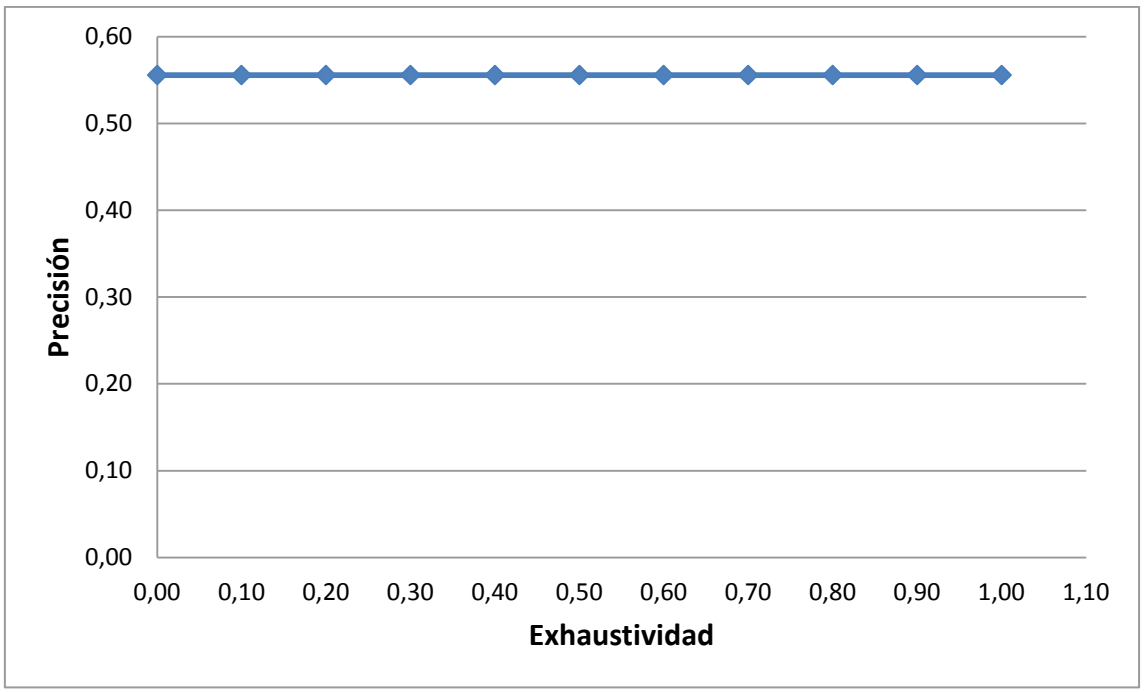
Anexo II. Curvas de resultados

En este anexo se presentan las 12 curvas de precisión-exhaustividad obtenidas como resultado del plan de pruebas del sistema y a partir de las cuales se ha obtenido la curva de precisión-exhaustividad media del sistema.

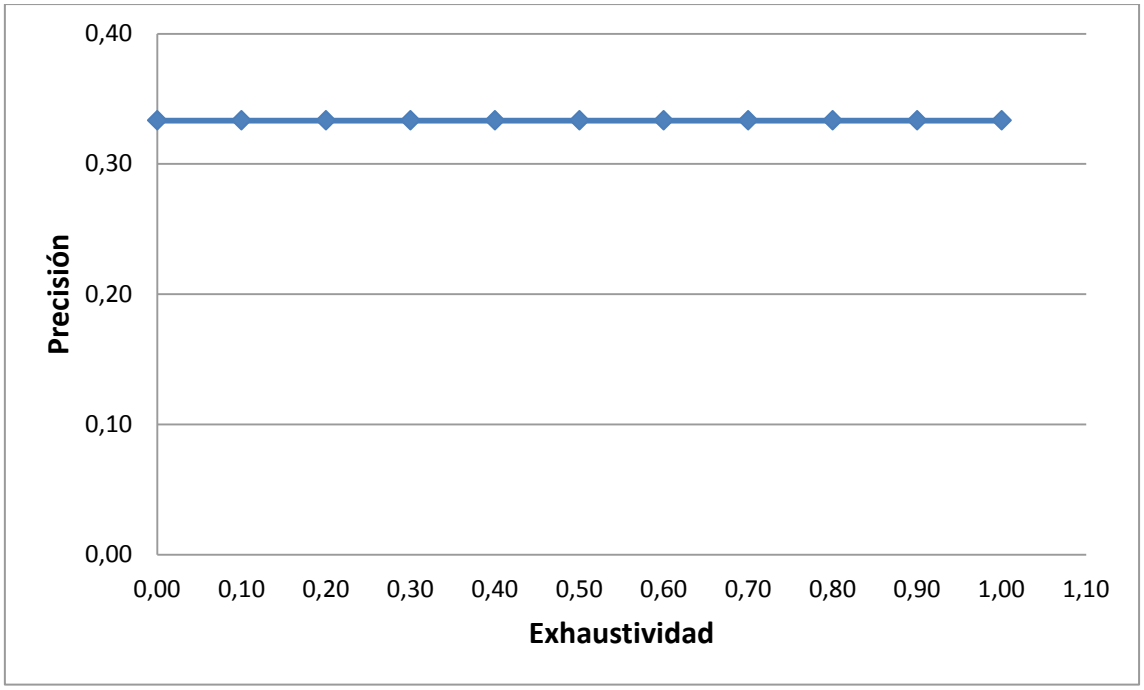
- Curva Precisión-exhaustividad resultante de la Demanda 1.



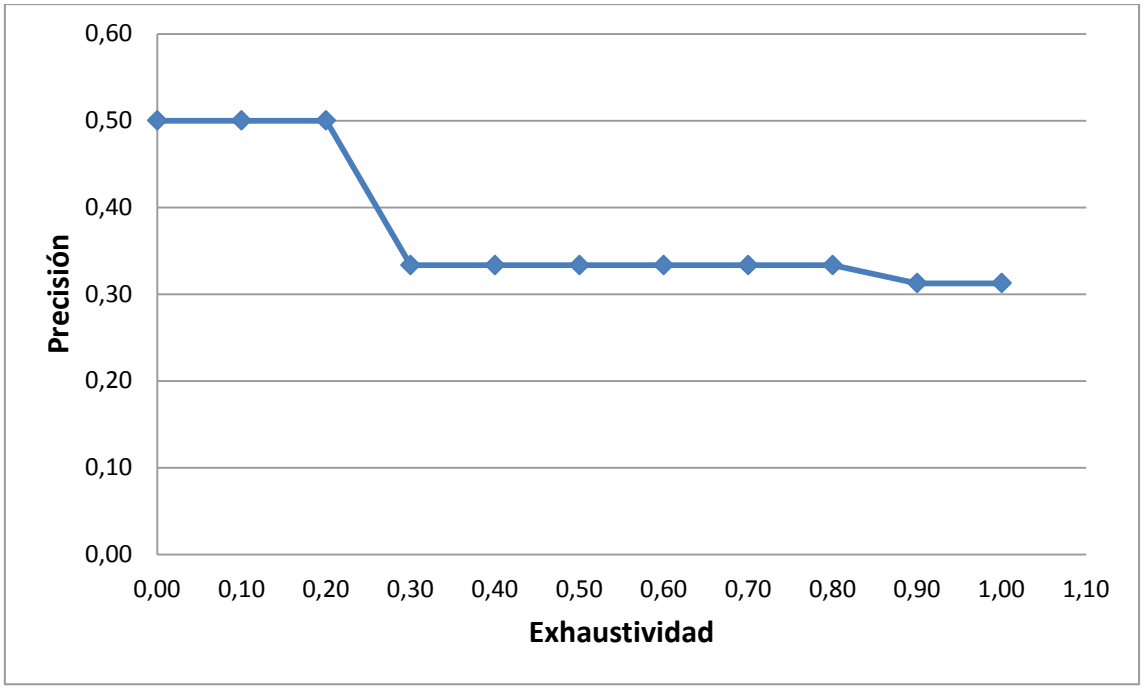
- Curva Precisión-exhaustividad resultante de la Demanda 2.



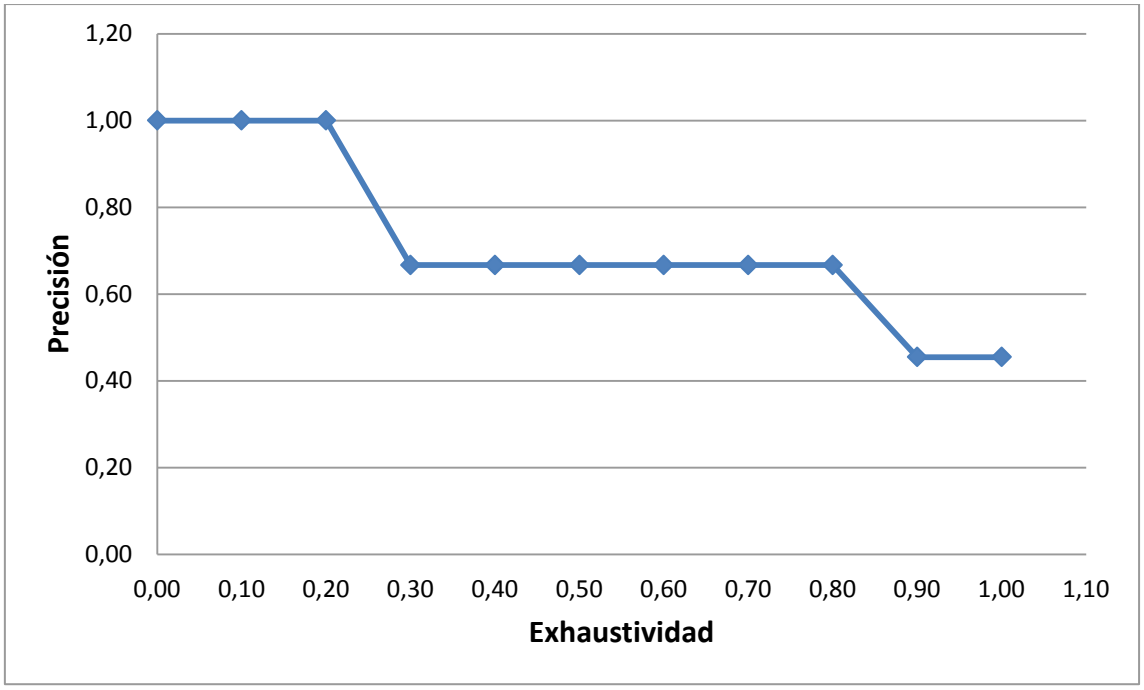
- Curva Precisión-exhaustividad resultante de la Demanda 3.



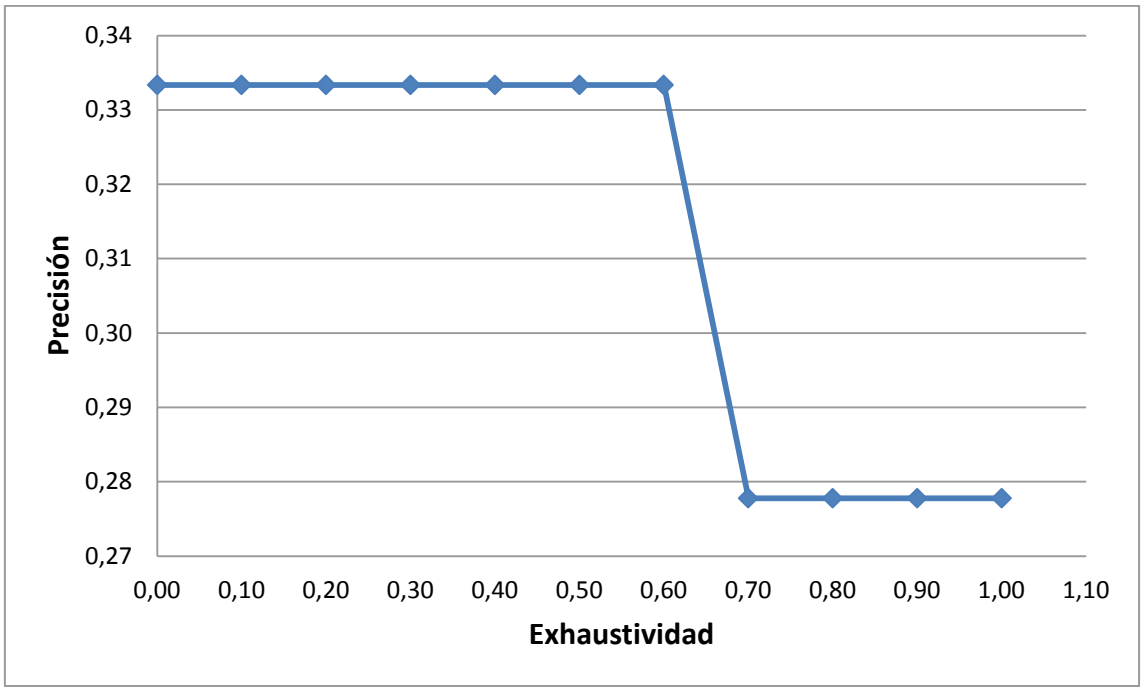
- Curva Precisión-exhaustividad resultante de la Demanda 4.



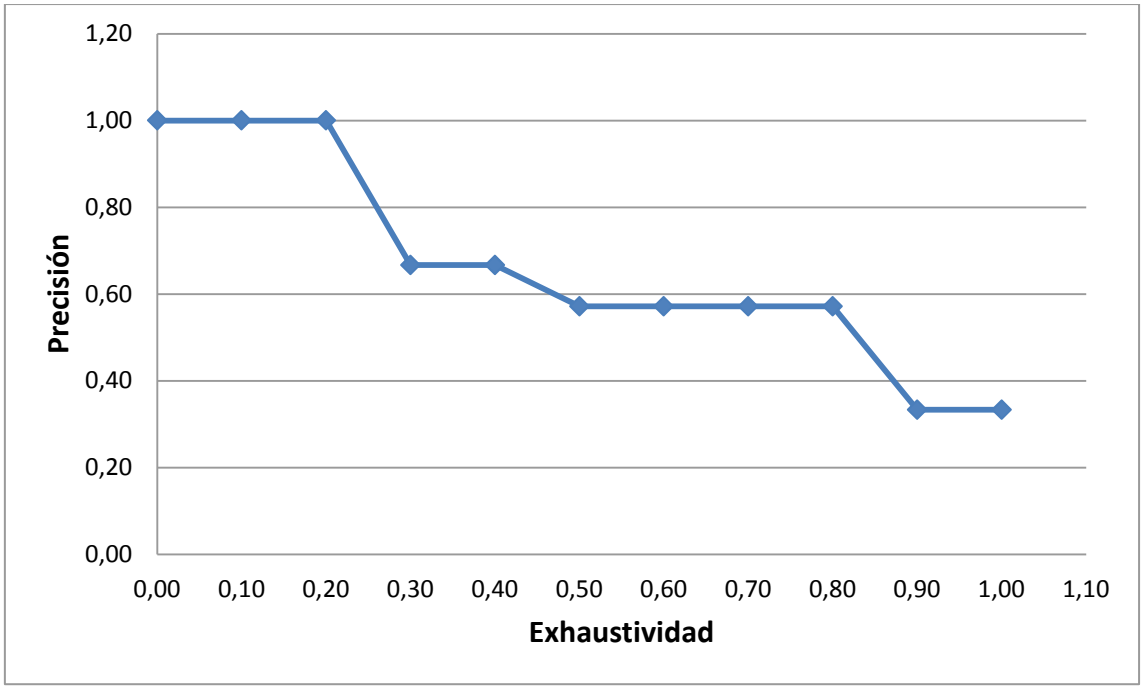
- Curva Precisión-exhaustividad resultante de la Demanda 5.



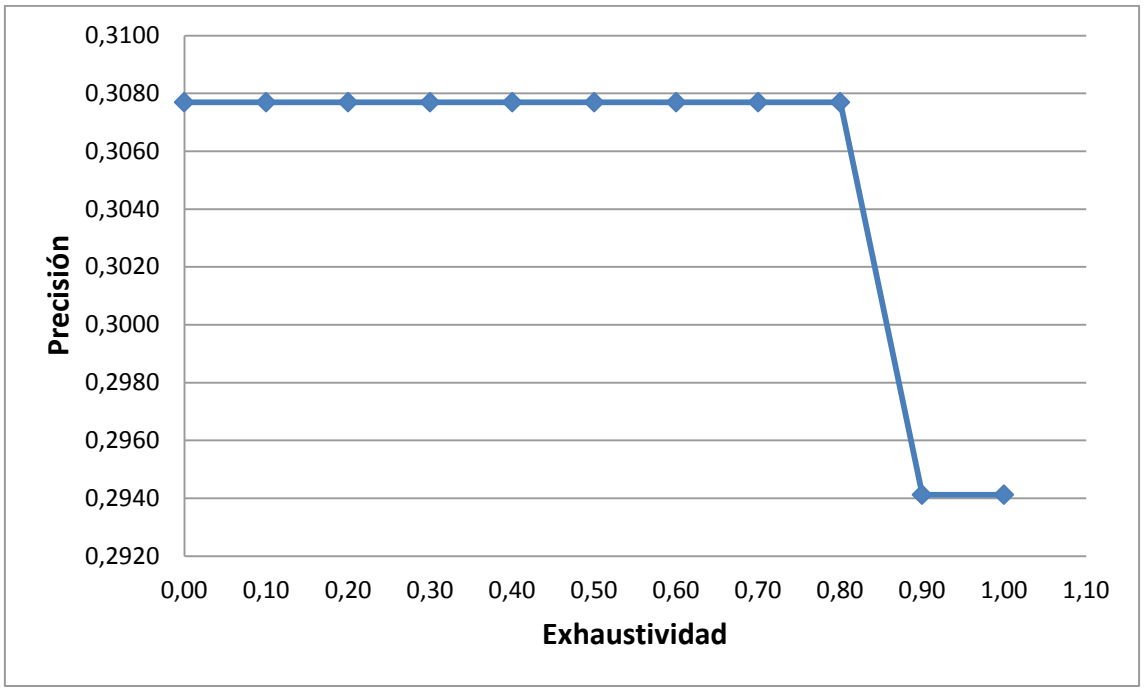
- Curva Precisión-exhaustividad resultante de la Demanda 6.



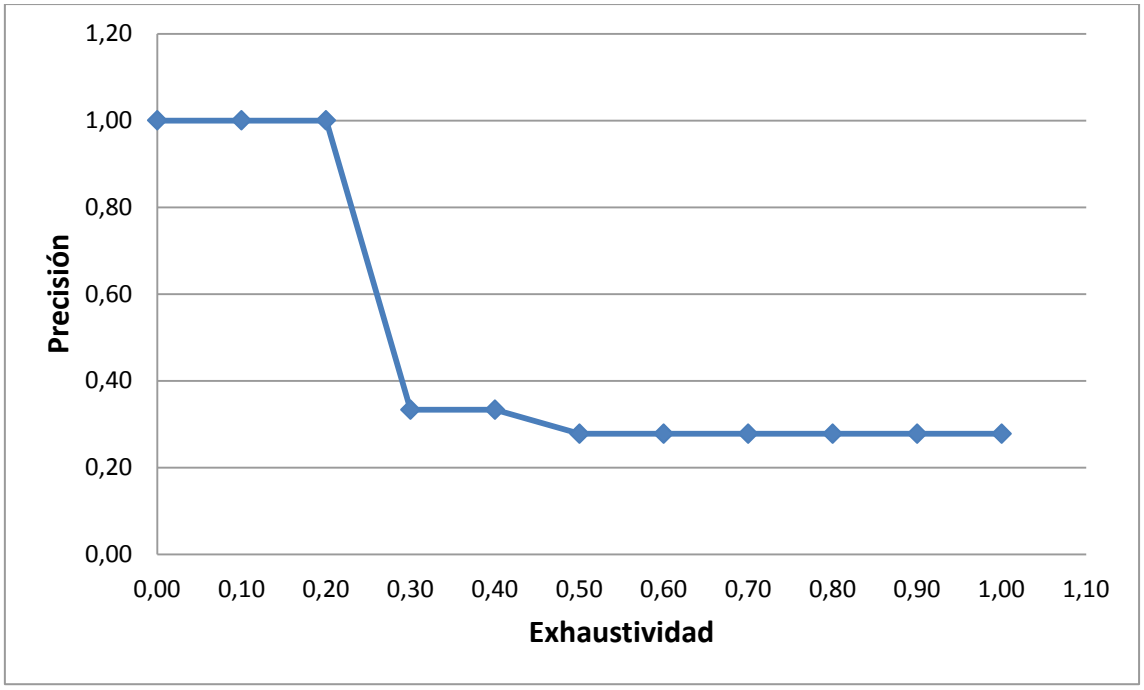
- Curva Precisión-exhaustividad resultante de la Demanda 7.



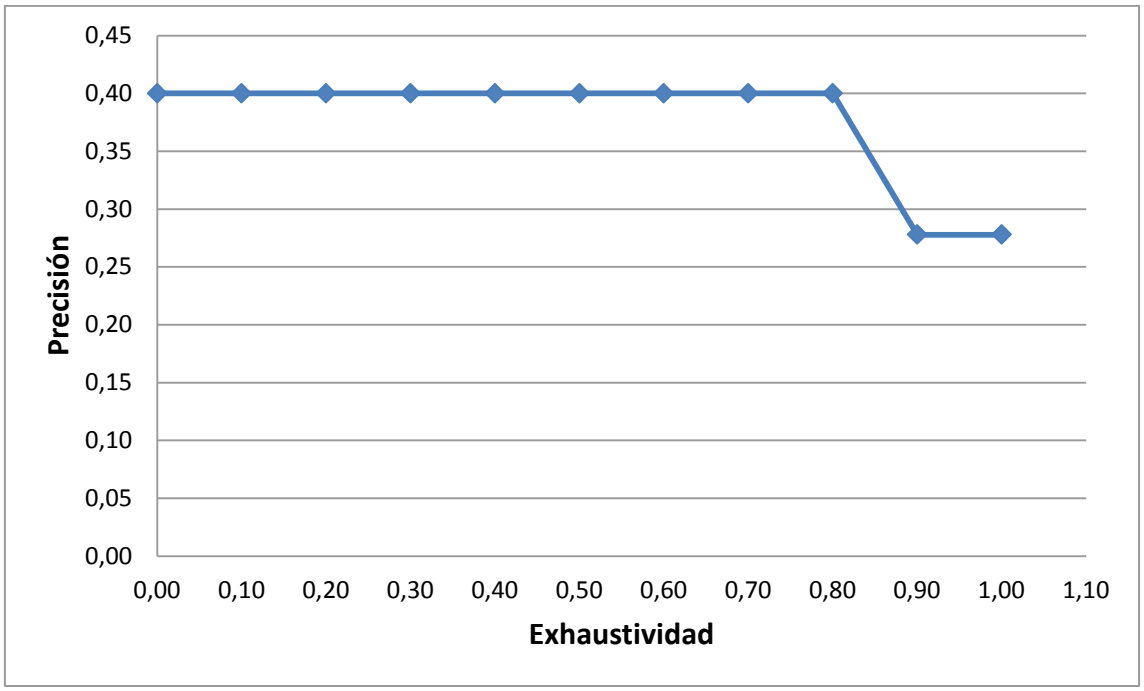
- Curva Precisión-exhaustividad resultante de la Demanda 8.



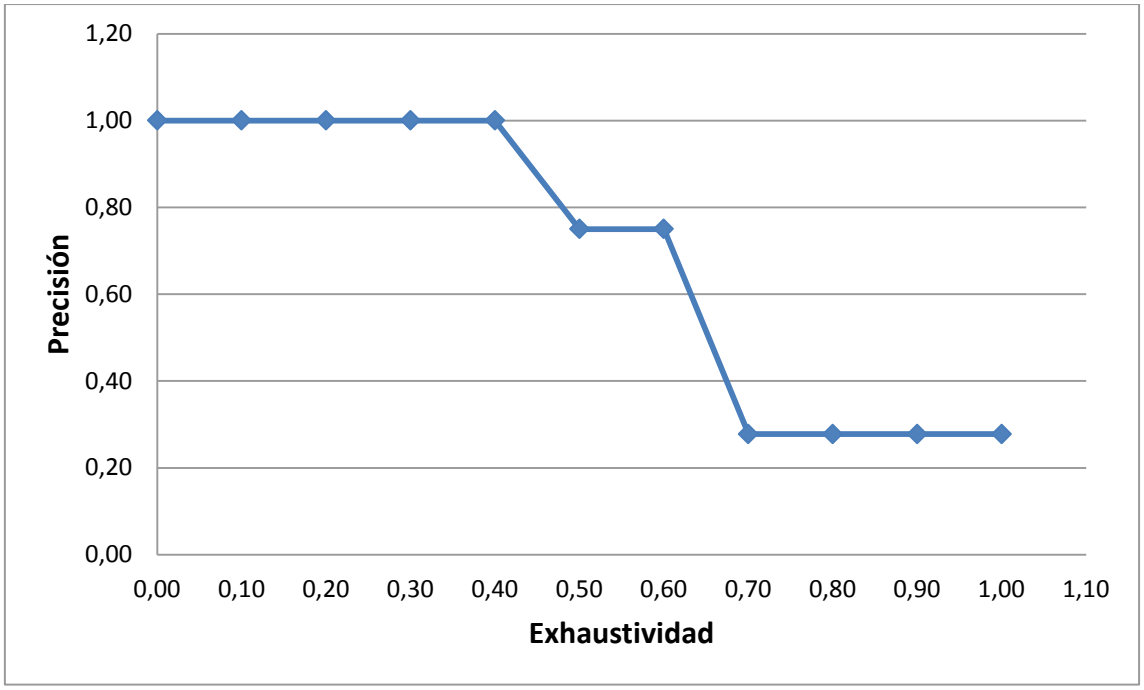
- Curva Precisión-exhaustividad resultante de la Demanda 9.



- Curva Precisión-exhaustividad resultante de la Demanda 10.



- Curva Precisión-exhaustividad resultante de la Demanda 11.



- Curva Precisión-exhaustividad resultante de la Demanda 12.